



الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieure et de la Recherche Scientifique



جامعة وهران 2 محمد بن أحمد
Université d'Oran 2 Mohamed Ben Ahmed

معهد الصيانة والأمن الصناعي
Institut de Maintenance et de Sécurité Industrielle

Département de Maintenance en Instrumentation

MÉMOIRE

Pour l'obtention du diplôme de Master

Filière : Génie Industriel

Spécialité : Ingénierie de la Maintenance en Instrumentation

Thème

ETUDE ET MISE EN ŒUVRE D'UN SYSTÈME DE COMPTAGE DE PIÈCES

Présenté et soutenu publiquement par :

GUEZGOUZ Mohammed

MESSILITI Zine Eddine

Devant le jury composé de :

Nom et Prénom	Grade	Etablissement	Qualité
TARI Nouria	MAA	IMSI	Président
HASSINI Abdellatif	MCA	IMSI	Examinateur
AISSANI Nassima	MCB	IMSI	Encadreur

Juin 2016

RESUME

Notre projet s'intitule:

« Etude et Mise En Œuvre d'Un Système De Comptage Des Pièces »

L'objectif est de réaliser un système de comptage de pièce, ce système est censé être placé sur un convoyeur d'une unité de production, pour comptabiliser le nombre de pièces manufacturées. Cette réalisation doit être le plus simple possible pour développer une maquette pédagogique de notre institut. Notre système est basé sur un microcontrôleur PIC16F628 qui a pour rôle : de détecter le passage des pièces grâce à un capteur photoélectrique composé d'un émetteur (diode laser) et un récepteur (LDR). Ensuite, assurer l'affichage via un multiplexage sur 4 afficheurs 7 segments. Enfin, communiquer ce nombre, à la demande, à un ordinateur par un port COM selon la norme de communication RS232. Toutes les étapes de conception, programmation et réalisations sont détaillées et illustrées dans ce mémoire.

Mots-clés: PIC16F628 - détecteur de passage – Afficheur 7 segment - affichage multiplexé - communication RS232.

ABSTRACT

Our Project is called:

« Study and Realization Of Pieces Counting System »

The objective is to achieve a piece counting system, this system is supposed to be placed on a conveyor of a production unit for counting the number of manufactured piece. This achievement should be as simple as possible to suit a pedagogical model in our institute. Our system is based on a PIC16F628 microcontroller whose role: to detect the passage of products through a photoelectric sensor consisting on a transmitter (laser diode) and a receiver (LDR). Next, to ensure displaying via 4 multiplexed 7-segment displays. Finally, communicate the number, on demand, to a computer via a serial port (COM port) with RS232 communication standard. All stages of design, programming and achievements are detailed and illustrated in this project report.

TABLE DES MATIERES

RESUME.....	I
DEDICACES	II
REMERCIEMENT	III
LISTE DES FIGURES.....	IV
INTRODUCTION GENERALE	1

CHAPITRE I : LES MICROCONTROLEURS ET L'INDUSTRIE

1.Introduction	2
2. Place des microcontrôleurs dans l'industrie moderne.....	2
3. Les microcontrôleurs	5
3.1 Généralité.....	5
3.2 Structure interne d'un microcontrôleur	5
3.2.1 Une unité centrale.....	5
3.2.2 Une mémoire	5
3.2.3 Une mémoire vive	5
3.2.4 Les ports d'entrées - sorties.....	6
3.2.5 Les bus internes	6
3.3 Architecture des Microcontrôleurs	6
3.4 Différentes Familles de microcontrôleurs.....	7
4. Les PIC	7
4.1 Définition d'un PIC	7
4.2 Les différentes familles des PIC	8
4.3 Identification d'un PIC	9
4.4 Structure d'un PIC	10
4.5 Structure minimale d'un PIC	10
4.6 Principe de fonctionnement.....	12
4.7 Alimentation des PIC	13
5. L'objectif de travail et problématique	14
5.1 Généralité sur les systèmes de comptage	14
5.2 Composition du Système.....	14
5.3 Problématique	15

CHAPITRE II : Etude des différentes parties du système de comptage

1. Introduction	17
2. L'alimentation	17
3. Le choix du microcontrôleur	17
4. PIC 16F628.....	18

TABLE DES MATIERES

4.1 Présentation	18
4.2 Architecture	19
4.2.1 Schéma fonctionnel	20
4.2.2 Unité centrale.....	20
4.2.3 Mémoire	21
4.2.4 Ports d'E/S.....	23
4.3 Les module	24
4.3.1 Les Timers	24
4.3.2 Comparateurs analogiques.....	24
4.3.3 Tension de référence	25
4.3.4 Module USART.....	25
4.4 Configuration de l'horloge	26
4.5 Interruptions.....	27
4.6 Watchdog	29
4.7 Le reset	29
5. Les afficheurs 7 segment.....	29
5.1 Afficheurs à anode commune	30
5.2 Afficheurs à cathode commune	30
6. Les divers détecteurs	31
6.1 Choix des détecteurs	31
6.2 Détecteurs photoélectrique	33
6.2.1 Barrière optique par transmission.....	33
6.2.2 Barrière à transmission avec réflecteur ou réflexion.....	34
7. Conclusion.....	34

CHAPITRE III : CONCEPTION ET PROGRAMMATION

1.Introduction	35
2. Implantation de schémas électriques du système de comptage.....	36
3. Commande multiplexé d'afficheurs 4 x 7 segments	37
4. La communication sérieRS232 entre PIC et PC	39
5. Réalisations de la barrière lumineuse	41

TABLE DES MATIERES

5.1 Emission du faisceau lumineux	42
5.2 Détection du faisceau lumineux	42
6. Partie Logiciel.....	43
6.1 Langage et compilateur mikroC for PIC	44
6.2 Isis Proteus Professional	49
7. Programme embarquée dans le PIC.....	50
7.1 Organigramme Principale	51
7.1.1 Organigramme d'interruption	52
7.1.2 Organigramme d'affichage multiplexé	52
7.2 Description du fonctionnement de système	53
7.3 Initialisation et configuration	53
7.4 Les fonctions et les variables.....	54
7.4.1 Fonction d'initialisation du module USART	54
7.5 Gestion d'affichage : 4 x 7 segments et dans l'application HyperTerminal.....	55
7.6 Interruption.....	56
8. Conclusion.....	56

CHAPITRE IV : REALISATION ET EXPERIENCES

1. Introduction.....	57
2. Programmeur Universel USB - Mini Pro TL866CS	57
3. Applications par le PIC 16f628.....	60
3.1 Commande d'un afficheur 7 segments.....	60
3.1.1 Le code C et les commentaires	60
3.1.2 Câblage de la plaque d'expérimentation	62
3.2 Test du module USART	62
3.2.1 Commentaire et code C	63
4. Réalisation de notre Projet :	66
4.1 Liste des composantes	66
4.2 câblage de la plaque d'expérimentation.....	66
5. Conclusion.....	68
Conclusion Générale	69
Bibliographie.....	70

LISTE DES FIGURES

CHAPITRE I

Figure I-1: Automate ou microcontrôleur	2
Figure I-2 : Schéma feux tricolores.....	4
Figure I-3 : Système de contrôle de température dans un four.	4
Figure I-4 : Détecteur de fumée.	4
Figure I-5 : Structure interne d'un microcontrôleur.....	6
Figure I-6 : L'architecture de Von Neumann et l'architecture de Harvard.	7
Figure I-7 : Structure de base d'un PIC.....	11
Figure I-8 : Les phases d'exécution d'instruction.....	12
Figure I-9 : Cycle d'instruction.....	12
Figure I-10: Schéma synoptique du système.....	15
Figure I.11 : Conception de l'unité de production	15

CHAPITRE II

Figure II-1 : Alimentation 9V suivie d'un régulateur 5V (7805).....	17
Figure II-2 : Brochage du pic 16F628.....	18
Figure II-3 : Architecture Harvard.	19
Figure II-4 : Schéma fonctionnel du PIC 16f628.....	20
Figure II-5 : Cœur d'un processeur PIC.	20
Figure II-6 : Mémoire programme.	21
Figure II-7 : Timer 0.....	24
Figure II-8 : Module de Tension de référence.....	25
Figure II -9 : Configuration d'oscillateur HS, XT et LP.	27
Figure II-10 Synoptique général des interruptions sur le 16F628.....	28
Figure II-11 : les modes du reset.....	29
Figure II-12 : Afficheur 7 segments	29
Figure II-13 : Schéma interne d'un afficheur à anode commune..	30
Figure II-14 : Schéma interne d'un afficheur à cathode commune	30

LISTE DES FIGURES

Figure II- 15 : L'algorithme de choix des détecteurs	32
Figure II-16 : Barrière par transmission	33
Figure II-17 : Barrière à transmission avec réflexion et réflecteur	34

CHAPITRE III

Figure III-1: Schémas électrique du système.	36
Figure III-2 : Brochage du 4511.....	37
Figure III-3 : le transistor 2N2222	39
Figure III-4 : RS232 broches de signal dans un connecteur femelle DB-9.....	39
Figure III-5 : Rôle du circuit max232	40
Figure III-6: Brochage du circuit max232 avec le PIC et PC	40
Figure III-7 : Schémas électrique de la barrière lumineuse.....	41
Figure III-8 : Diode laser.....	42
Figure III-9 : Photorésistance.	42
Figure III-10 : L'environnement IDE du compilateur MICROC PRO	45
Figure III-11 : Avertissement des erreurs	49
Figure III-12 : Simulation avec Proteus..	50

CHAPITRE IV

Figure IV-1 : Programmeur Universel USB - Mini Pro TL866CS.	57
Figure IV-2 : L'IDE MiniPro V6.00	57
Figure IV- 3 : Simulation de la commande d'afficheur par ISIS	60
Figure IV- 4 : Le montage sur la plaque d'expérimentation d'afficheur 7 segments	62
Figure IV- 5 : La liaison série RS232 entre PIC et PC	62
Figure IV- 6 : Configuration des paramètres d'HyperTerminal.....	63
Figure IV- 7 : Schémas de la liaison RS232 et Simulation	64
Figure IV-8 : Montage de système dans la plaque d'expérimentation.....	66

LISTE DES FIGURES



Introduction Générale

INTRODUCTION GENERALE

L'envolée de l'informatique industrielle débute en 1971, année qui marque la fabrication du premier microprocesseur par la société Intel. Ce microprocesseur, le '4004', comportait déjà 2300 transistors et fonctionnait avec un bus de 4 bits. Depuis, l'intégration du nombre de transistors dans les microprocesseurs n'a cessé d'évoluer, parallèlement à la puissance de calcul et à la rapidité d'exécution.

Aujourd'hui, la plupart des réalisations électroniques dans de nombreux secteurs de l'industrie font appel à ces composants qui peuvent gérer n'importe quel automatisme. L'avènement des microcontrôleurs, qui associent au microprocesseur de base un programme intégré au circuit, ainsi que des périphériques et de la RAM (Random Access Memory), a permis de faire évoluer les montages vers encore plus de simplicité et de rapidité, les périphériques étant intégrés au circuit. Les microcontrôleurs sont aujourd'hui implantés dans la plupart des réalisations grand public ou professionnelles. On les trouve dans les télécommandes, les téléphones, les machines à laver, les perceuses, et à profusion dans les automobiles.

Dans le cadre de se familiariser avec les microcontrôleurs notre travail a pour objectif la réalisation d'un système de comptage des pièces avec le PIC 16f628.

Notre travail sera organisé selon une méthode de travail bien déterminée :

- 1- Etude théorique des différents composants de notre système.
- 2- Conception et simulation du système.
- 3- Réalisation pratique et diagnostique.

Chapitre

I

**Les microcontrôleurs
Et
L'industrie**

1. Introduction

Les microcontrôleurs sont très utilisés dans le monde de l'industrie, notamment dans les systèmes embarqués. On pourra donc les retrouver dans l'aéronautique, l'aérospatial, l'automobile, l'électronique grand public. Leur polyvalence, et leur taille les rendent intéressants pour des modules de traitement de données numériques et aussi analogiques. Ils sont certes peu puissants comparés à des processeurs dédiés, mais ils les compensent par leur prix mais surtout leur taille. D'ailleurs, la puissance de ses derniers n'est pas toujours nécessaire pour leur utilisation.

2. Place des microcontrôleurs dans l'industrie moderne

Comme leur nom l'indique, ces circuits effectuent des tâches de contrôles. Un exemple typique consiste à gérer des signaux provenant du monde extérieur. Ces signaux peuvent provenir de capteurs analogiques ou numériques et sont connectés aux pattes d'entrées du microcontrôleur. Chaque entrée représente une part d'information sur l'état d'un dispositif externe. Les sorties servent de commandes pour ce dispositif, qui peut être composé de relais, de moteurs, etc. Le rôle du microcontrôleur est d'analyser les entrées et d'affecter les bonnes valeurs en sortie en fonction de l'état courant du système, à la manière d'un automate d'états. Dans les systèmes plus larges, les microcontrôleurs sont aussi utilisés pour la configuration de périphériques et de coprocesseurs, la signalisation et la synchronisation des différents éléments du système^[1].



Figure I-1: Automate ou microcontrôleur.

Le principal rôle des microcontrôleurs est donc de prendre des décisions en fonction de l'état d'un système et de la valeur de ses entrées. Leur architecture est spécialement conçue pour gérer efficacement les branchements conditionnels, les interruptions, les changements de contexte, les tests au niveau bit et les hiérarchies mémoires complexes.

Leur capacité en terme de calcul arithmétique est faible : leur chemin de données est généralement composé d'un banc de registres à usage général couplé à une unité arithmétique et logique (ALU) capable d'effectuer les opérations élémentaires que sont l'addition, la soustraction et les fonctions logiques de base. Les performances de ces circuits en traitement du signal sont faibles, pour deux principales raisons :

- 1- Les données et les instructions utilisent le même bus mémoire. La bande passante disponible pour accéder aux données est donc faible : le filtre FIR nécessiterait 2 données pour l'exécution convenable d'une multiplication par cycle, alors que l'architecture Von Neumann (un bus de données et un bus d'adresses) des microcontrôleurs ne peut fournir qu'une donnée tous les 2 cycles.
- 2- Les opérateurs arithmétiques sont peu performants. Les opérations de multiplication largement utilisées en traitement du signal sont soit émulées par logiciel en l'absence d'opérateur câblé spécialisé, soit réalisés sous forme d'opérateurs pipelines. Dans tous les cas, une multiplication prend donc plusieurs cycles. Il en va de même pour les opérations de saturation ou d'arrondi, indispensables pour la gestion de la précision lorsqu'on travaille en virgule fixe ^[1].

Les microcontrôleurs à basse consommation sèment la discorde dans l'industrie des semi-conducteurs. A Electronica, le grand salon des composants et sous-systèmes électroniques qui réunit plus de 2650 exposants à Munich (Allemagne), du 13 au 16 novembre 2012, les fournisseurs se livrent à une véritable bataille de communication dont l'enjeu est de convaincre les utilisateurs qu'ils détiennent la palme de sobriété sur le marché. Selon Freescale, ils représentent un marché de l'ordre de 15 milliards de dollars dans le monde, dont 8,5 milliards de dollars hors automobile et cartes à puces ^[2].

Les microcontrôleurs restent très utilisés pour des systèmes peu complexes, nous trouvons des applications dans tous les domaines de la vie quotidienne.

Voici des exemples d'utilisation des microcontrôleurs : l'urbanisme (feu tricolores), Commercial (électroménager...), Automobile (ABS, tableau de bord, contrôle des sièges...), Télécommunications (cartes FAX et MODEM, Minitel, téléphones portables...) et dans l'industrie (réalisation des automates programmables, contrôle de processus divers, supervision...).

Exemple des feux tricolores :

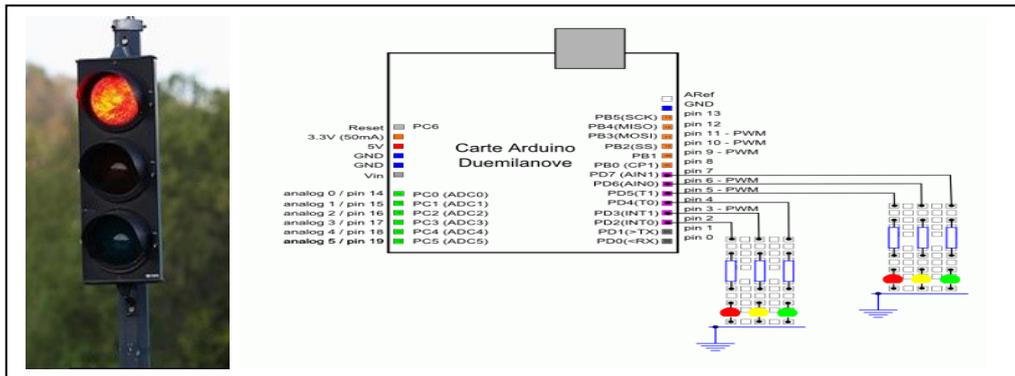


Figure I-2 : Schéma feux tricolores.

En milieu industriel :

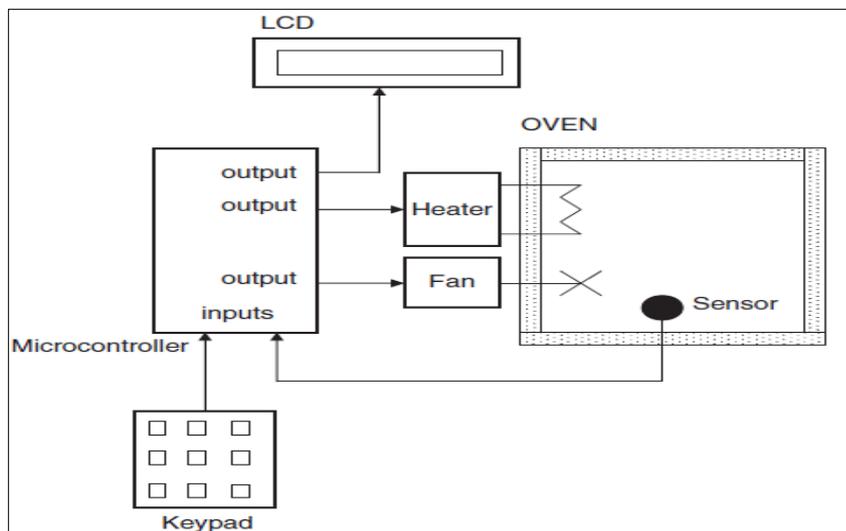


Figure I-3 : Système de contrôle de température dans un four.



Figure I-4 : Détecteur de fumée

3. Les microcontrôleurs ^[11]

3.1 Généralité

C'est un ordinateur monté dans un circuit intégré. Les avancées technologiques en matière d'intégration, ont permis d'implanter sur une puce de silicium de quelques millimètres carrés la totalité des composants qui forment la structure de base d'un ordinateur. Comme tout ordinateur, on peut décomposer la structure interne d'un microprocesseur en trois parties :

- Le processeur
- Les mémoires
- Les périphériques

3.2 Structure interne d'un microcontrôleur :

Un microcontrôleur le plus simple qu'il soit, possède au minimum les éléments suivants :

3.2.1 Une unité centrale:

Qui est le cœur du système, également appelé CPU pour Central Processing Unit, dans cette unité centrale nous retrouverons plusieurs éléments telle que l'unité arithmétique et logique (UAL).

3.2.2 Une mémoire:

Contenant le programme à exécuter par le microcontrôleur, généralement appelée mémoire morte ou ROM (Read Only Memory), mémoire à lecture seule. Cette mémoire a la particularité de sauvegarder en permanence les informations qu'elle contient, même en absence de tension (ce qu'est primordiale, sinon il faudrait reprogrammer le microcontrôleur à chaque remise sous tension).

3.2.3 Une mémoire vive:

Egalement appelée RAM (Random Access Memory), cette mémoire permet de sauvegarder temporairement des informations. Il est à noter que le contenu d'une RAM n'est pas

sauvegardé que pendant la phase d'alimentation du circuit. Le microcontrôleur pourra utiliser cette mémoire pour stocker des variables temporaires ou faire des calculs intermédiaires.

3.2.4 Les ports d'entrées - sorties:

Permettant au microcontrôleur de dialoguer avec l'extérieur pour par exemple prendre l'état d'un capteur, d'un interrupteur ou bien pour allumer une led ou piloter un relais.

3.2.5 Les bus internes:

Permettent la communication entre les différents éléments intégrés au microcontrôleur. Il existe trois sortes de bus (Bus de données, adresse et de commande).

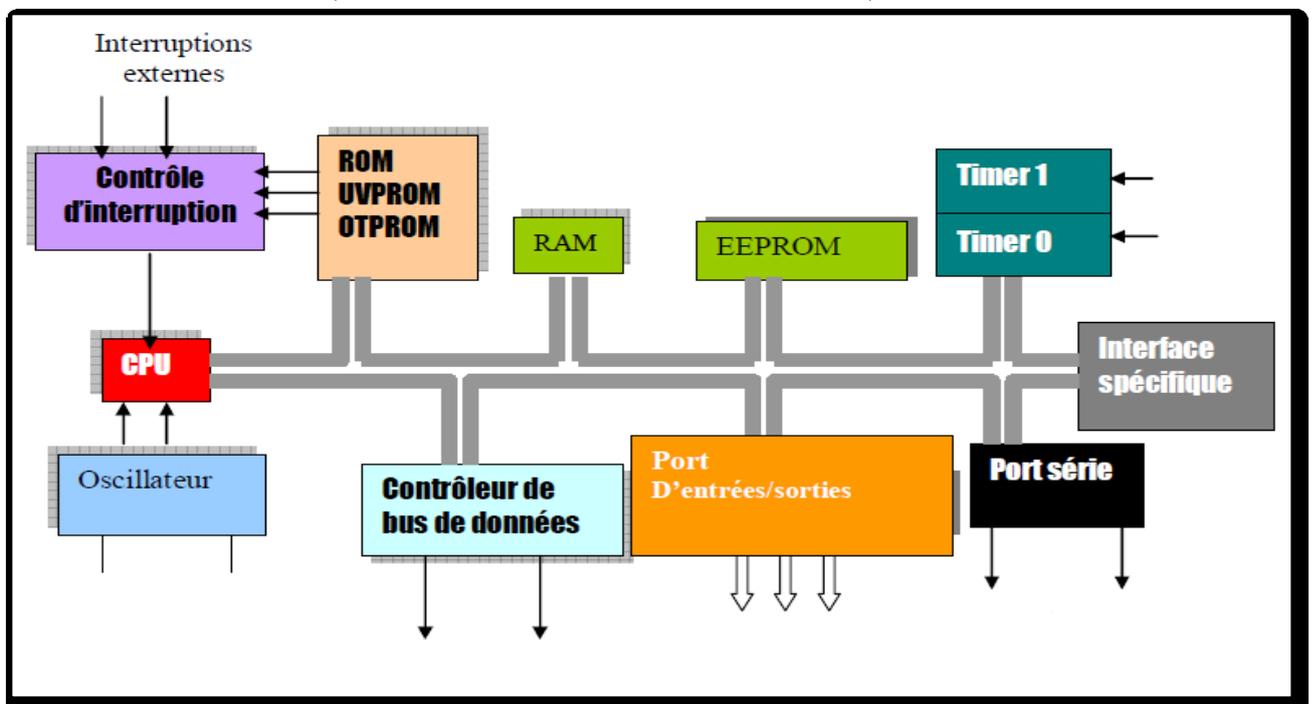


Figure I-5: structure interne d'un microcontrôleur.

3.3 Architecture des Microcontrôleurs

Deux types classiques d'architectures dans les microcontrôleurs (Figure 6). **Architecture de Von Neumann**, est le plus utilisé dans les microcontrôleurs, tout l'espace mémoire sur le même bus, les instructions et les données utilisent simultanément le même bus. Dans **l'architecture de Harvard** (utilisé par les microcontrôleurs PIC), la mémoire d'instruction et la mémoire des données sont sur bus séparés, ce qui leur permet d'être récupérés en même temps, résultant en une amélioration de la performance ^[11].

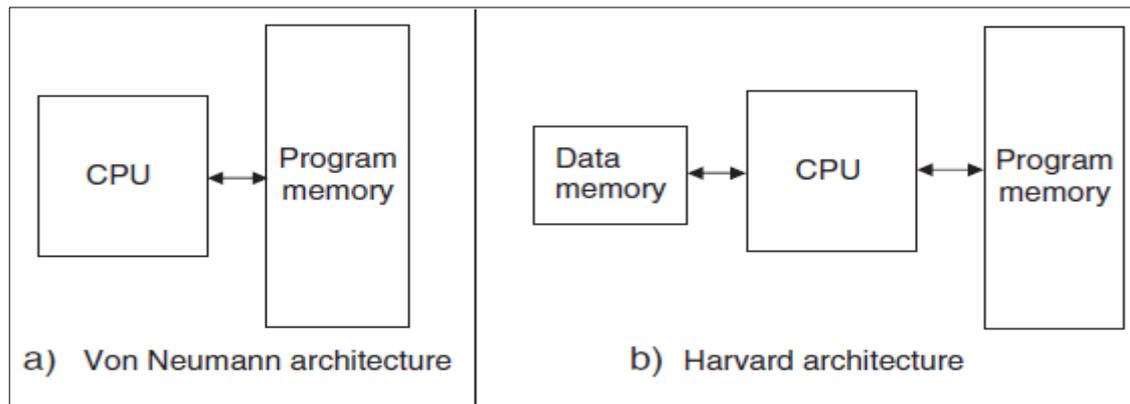


Figure I-6 : L'architecture de Von Neumann et l'architecture de Harvard.

3.4 Différentes Familles de microcontrôleurs :

On trouve plusieurs familles des microcontrôleurs citons :

- La famille Atmel AT91.
- La famille Atmel AVR.
- Le C167 de Siemens.
- La famille Hitachi H8.
- La famille des PIC de Microchip.
- La famille des DSPIC de Microchip.
- La famille PSOC de Cypress.
- La famille LPC21xx ARM7-TDMI de Philips.
- La famille V800 de NEC.
- La famille K0 de NEC.

4. Les PIC

4.1 Définition d'un PIC:

Un PIC est un microcontrôleur, c'est à dire une unité de traitement d'information de type microprocesseur à laquelle on a ajouté des périphériques internes permettant de faciliter l'interfaçage avec le monde extérieur sans nécessiter l'ajout de composants externes^[3].

La société américaine Microchip Technologie a mis au point dans les années 1980 un microcontrôleur CMOS (Complementary Metal-Oxide Semi-conductor), le PIC. Ce composant, encore très utilisé à l'heure actuelle, est un compromis entre simplicité d'emploi, rapidité et prix de revient^[6].

La dénomination PIC est sous copyright de Microchip, donc les autres fabricants ont été dans l'impossibilité d'utiliser ce terme pour leur propre microcontrôleur.

On trouve sur le marché 2 familles opposées, les RISC et les CISC (Complex Instructions Construction Set). Chez les CISC, on diminue la vitesse de traitement, mais les instructions sont plus complexes, plus puissantes, et donc plus nombreuses. Il s'agit donc d'un choix de stratégie.

Les PIC sont des composants RISC (Reduced Instructions Set Computing) ou encore composant à jeu d'instructions réduit.

L'avantage est que plus on réduit le nombre d'instructions, plus facile et plus rapide en est le décodage, et plus vite le composant fonctionne^[3].

4.2 Les différentes familles des PIC :^[4]

Il n'existe plus aujourd'hui que trois grandes familles de microcontrôleur PIC. Si l'on s'en tient à la classification adoptée par Microchip depuis mi-2004, on peut les présenter de la façon suivante selon la documentation Microchip :

- **La famille Base-Line:**

Elle ne contient que quelques circuits mais, paradoxalement, ceux-ci ont un avenir totalement différent selon leurs références.

On y trouve en effet les petits 12C508, 12C509, 12F508 et 12F509 dont l'unité centrale ne travaille que sur 12 bit, ainsi que les encore plus petits et très récents circuits de la famille 10Fxxx. Ces derniers sont baptisés par Microchip les plus petits microcontrôleurs du monde. Ils sont en effet disponibles en boîtier en montage en surface CMS. Ce sont des Pic qui s'utilisent comme leurs homologues plus encombrés pour peu que l'on puisse se satisfaire de leur faible nombre de lignes d'entrées/sorties.

Les inconvénients de ces circuits sont la capacité de leurs mémoires, relativement faible et mal adaptés à une programmation en langage évolué. Figurent aussi dans cette gamme Baseline les vieux 16C5x, toujours à unité centrale 12 bits et ne possèdent aucune particularité qui font le succès actuel des PIC. Ces circuits sont condamnés à disparaître à plus ou moins longue échéance et, quoi qu'il en soit, sont vivement déconseillés pour toute réalisation nouvelle.

- **La famille Mid-Range:**

Celle qui comprend actuellement le plus de références. Ces circuits prennent évidemment les concepts qui ont fait le succès de la vieille famille 16C5x. Ils utilisent la même architecture et le même jeu d'instruction de base mais disposent de ressources internes plus nombreuses avec, par exemple des ports séries variés. Enfin de nombreux circuits de cette famille sont disponibles avec de la mémoire de programme type flash. Cette mémoire est programmable et effaçable électriquement et permet donc très facilement de réaliser des maquettes ou des prototypes. Ils sont à la programmation en langage C ainsi disposent de mémoires de travail et de programme de grande capacité.

- **La famille High-Performance:**

Cette famille ne contient plus aujourd'hui que les circuits dont les références sont de type 18Cxxx ou 18Fxxx selon que de leur mémoire de programme est de type OTPROM ou flash. Compte tenu de leur richesse en ressources matérielles internes mais aussi des capacités très importantes de leurs mémoires vives et de leurs mémoires de programme, ces circuits sont les mieux adaptés à la programmation en langage C. Comme ils sont en fait une évolution logique des circuits des familles précédentes dont ils prennent les éléments essentiels, la transformation d'une application prévue pour un PIC 16fxxx en une application à base 18fxxx ne pose aucun problème.

4.3 Identification d'un PIC^[3] :

Pour identifier un PIC, on utilise simplement son numéro 'xx(L)XX yy -zz':

- xx : indiquent la catégorie du PIC, par exemple : 16 indique un PIC Mid-Range, 18 indique PIC high-Range.
- L : (Vient ensuite parfois une lettre L) celle-ci indique que le PIC peut fonctionner avec une plage de tension beaucoup plus tolérante.
- XX : Type de mémoire de programme (C : indique que la mémoire programme est un EPROM ou EEPROM, CR : indique le type de mémoire, CR(ROM) et F (FLASH). À ce niveau, on rappelle que seule une mémoire FLASH ou EEPROM est susceptible d'être effacée.
- yy : Identification.
- zz : représente la fréquence d'horloge maximale que le PIC peut recevoir, par exemple -04 pour un 4MHz.

4.4 Structure d'un PIC ^[6]

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 registre et périphériques possédant des caractéristiques différents.

Les différences de caractéristique 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.

Les data book des fabricants renseignent l'utilisateur sur les points cités ci-dessus.

4.5 Structure minimale d'un PIC :

La structure minimale d'un PIC (Figure I-7) 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. La capacité de carte RAM est variable selon les PIC.
- Une Unité Arithmétique et logique (UAL ou ALU en anglais) est chargée d'effectuer toutes les opérations arithmétique de basse (addition, soustraction, 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 commande un relais (sortie logique).

- Un registre compteur de programme (CP ou PC en anglais) est chargé de pointer l'adresse mémoire courante de l'instruction à réaliser par le microcontrôleur. Le contenu du registre PC évolue selon le pas de programme.

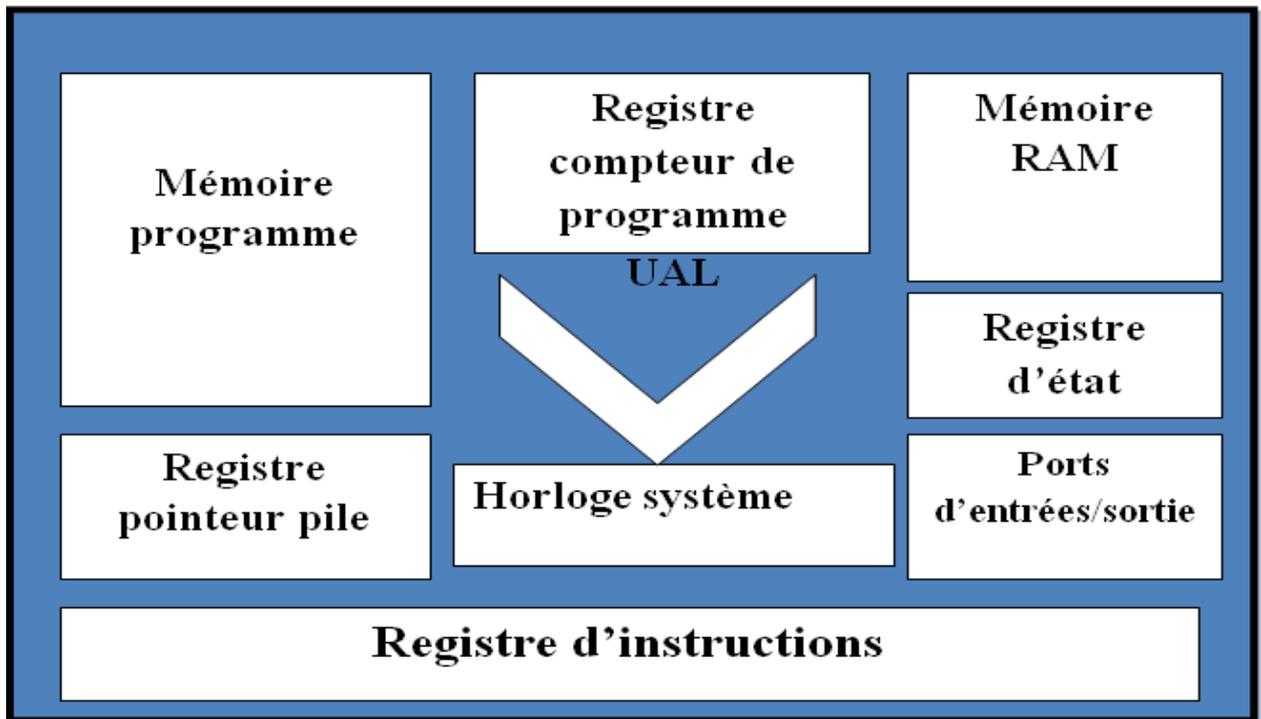


Figure I-7 : Structure de base d'un PIC.

- 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.
- 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^[6].

4.6 Principe de fonctionnement ^[3]

Un PIC exécute des instructions. On définit « le cycle instruction » comme le temps nécessaire à l'exécution d'une instruction. Une instruction est exécutée en deux phases :

1. La phase de recherche du code binaire de l'instruction stocké dans la mémoire de programme.
2. La phase d'exécution ou le code de l'instruction est interprété par le processeur et exécuté.

Chaque cycle instruction dure 4 coup d'horloge comme le montre la figure 4 :

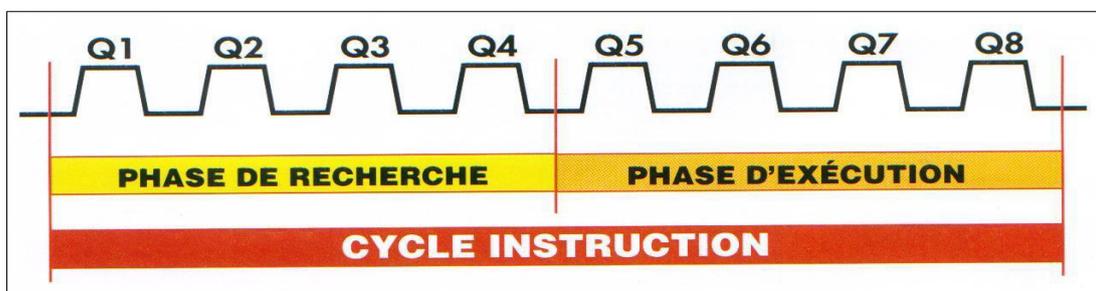


Figure I-8 : Les phases d'exécution d'instruction.

On pourrait donc croire qu'un cycle instruction dure 8 cycles d'horloge mais l'architecture Harvard du PIC lui permet de réduire ce temps par deux. En effet, comme les instructions issues de la mémoire de programme circulent sur un bus différent de celui sur lequel circulent les données, ainsi le processeur peut effectuer la phase de recherche d'une instruction pendant qu'il exécute l'instruction précédente (Figure 9).

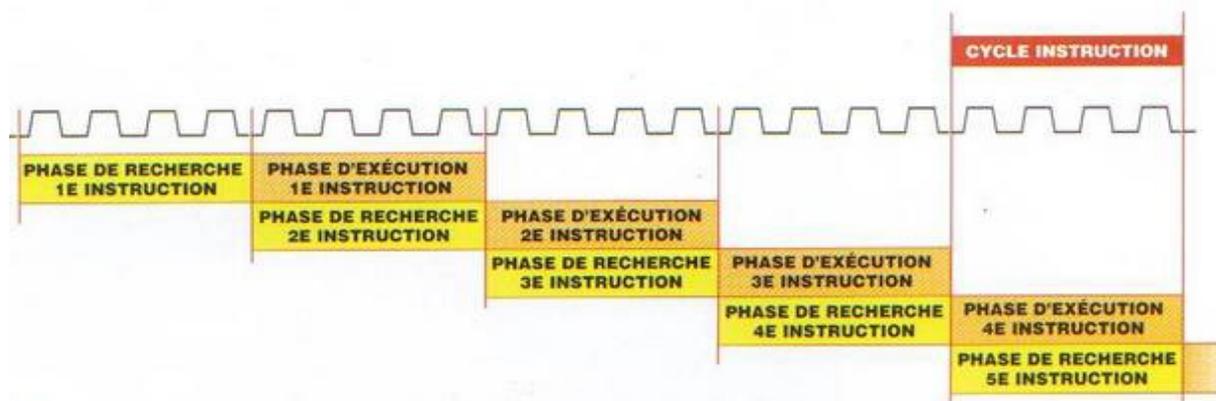


Figure I-9 : Cycle d'instruction.

4.7 Alimentation des PIC :

La majorité des PIC, toutes familles confondues, existent avec diverses options de tension d'alimentation.

Ici encore, c'est en principe une lettre centrale de la référence qui permet de savoir dans quelle situation on se trouve mais, en cas de doute, il est tout de même conseillé de vérifier ce qu'il en est dans la fiche technique du circuit que vous aurez retenu.

Ce principe de codage fonctionne théoriquement de la façon suivante :

- Les circuits ne disposant d'aucune lettre centrale particulière, autre que celle de désignation du type de mémoire vue précédemment, fonctionnent uniquement dans la plage de tension qui s'étend de 3 ou 4 volts selon les références à 5.5volts.
- Les circuits affectés de la lettre « L » précèdent la lettre type de mémoire, fonctionnent dans une plage de tension étendue qui dépend du type de mémoire de programme dont est muni le circuit. Selon les référence, cette tension minimum de fonctionnement peut parfois descendre jusqu'à 2 volts (16LF87x par exemple mais il n'est pas le seul dans ce cas).

Dans tous les cas, les entes/sortie des microcontrôleurs sont compatibles TTL lorsque ces derniers sont alimentés par une tension comprise dans la plage d'alimentation normalisée des circuits TTL, soit de 4,75 à 5,25 volts.

Dans le cas contraire, elles sont compatibles avec les niveaux logiques des circuits CMOS alimentés sous la même tension que le microcontrôleur.

Enfin, notez que la mémoire vive ou RAM contenue dans les microcontrôleurs PIC est capable de conserver son contenu lorsque le circuit est placé en mode sommeil et ce jusqu'à une tension d'alimentation aussi basse que 1,5 volts.

Dernière précision, mais qui va quasiment de soi en présence de circuits CMOS : le positif de la tension d'alimentation est appliqué à la ou aux pattes VDD du boîtier et la masse à la ou aux pattes VSS.

Lorsque plusieurs pattes VDD et VSS sont présentes sur un même boîtier, elles doivent toutes être reliées en externe à l'alimentation, et ce bien qu'elles soient parfois interconnectées en interne dans certains circuit. En outre, ces pattes d'alimentation doivent être découplées au plus près du circuit au moyen d'un condensateur de deux à 100 nF de type céramique^[4].

5. L'objectif de travail et problématique

5.1 Généralité sur les systèmes de comptage : ^[5]

D'une façon générale, on appelle compteur un système destiné à visualiser une grandeur, une quantité dans une unité précise. Il s'agit souvent de quantifier des grandeurs discrètes (numériques).

Les systèmes de comptage sont très répandus dans la vie courante. Ils assurent automatiquement une gestion efficace des entrées et sorties de personnes ou d'objets dans les lieux publics et les complexes industriels.

Nous pouvons citer quelques applications parmi les plus importantes :

- Le dénombrement des personnes occupant un magasin.
- Le contrôle d'occupation d'un parking.
- Gérer un flux de produits dans une chaîne de production.

Le système que nous sommes en vue de proposer a pour fonction d'afficher le nombre des objets passent sur un tapis convoyeur de 0 jusqu'aux 9999.

5.2 Composition du Système :

En effet, on trouve :

- L'unité de traitement et de contrôle qui est le microcontrôleur.
- L'affichage de nombres.
- L'unité de détection du passage.
- Unité d'alimentation stable.

Ensuite, on donne le schéma synoptique de notre système :

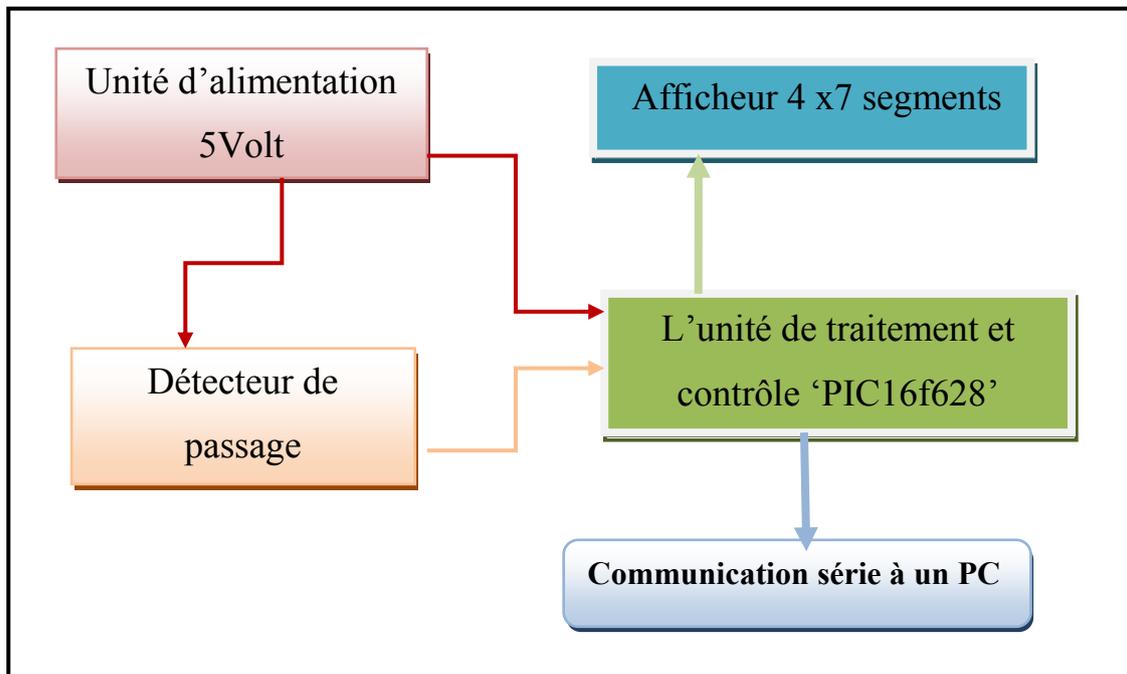


Figure I-10: Schéma synoptique du système.

5.3 Problématique :

Soit un tapis convoyeur roulant qui déplace des pièces dans l'unité de production. Et afin de calculer le nombre de pièces fabriquées, un détecteur de passage sera placé est lié à un microcontrôleur, ainsi un afficheur pour présenter le nombre de pièces.

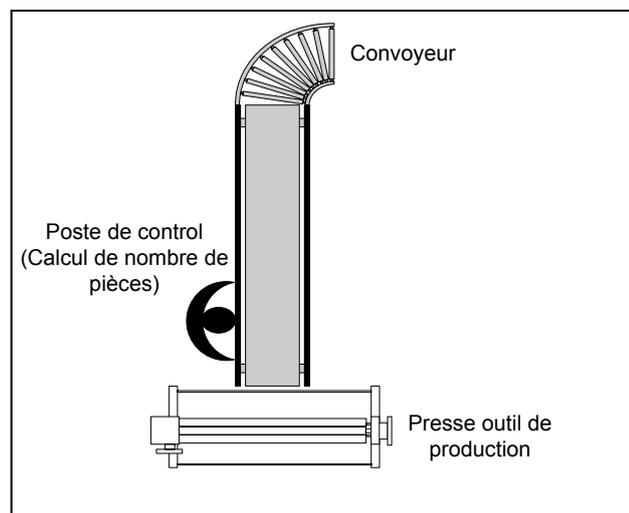


Figure I-11 : Conception de l'unité de production

Le but de notre projet est de réaliser le comptage et l'affichage du nombre des objets passer de 0 à 9999. Lorsqu'une pièce passe par le détecteur un signal est envoyé au système d'affichage qui est incrémenté par 1.

Travail demandé :

- Etudier les capteurs pour sélectionner celui qui pourra être utilisé sur notre système, à savoir détecter le passage des pièces de moyenne est petite taille.
- Etudier les afficheurs et sélectionner celui qui affichera le nombre de pièces traités dans le système.
- Etudier les microcontrôleurs et choisir celui qui commandera l'affichage en utilisant le signal du détecteur.

Tout ceci en optimisant le cout du système.

Réaliser le programme de commande, le simuler, l'implémenter et le déployer sur la maquette d'essai.

Chapitre

II

**Etude des différentes
Parties du système
De comptage**

1. Introduction

Dans un système de comptage, il est indispensable d'avoir une unité de traitement des informations, détecteur de passage, un afficheur pour faire apparaître le nombre des objets compté et une alimentation continue.

Bien que ce chapitre ait pour thème l'étude des différents composants de notre système de comptage, il ne saurait être question de se lancer dans une telle étude sans faire quelques rappels quant aux différents circuits disponibles, puisque leurs particularités peuvent conduire à les choisir ou les éliminer dans le cadre de notre projet.

2. L'alimentation

Dans le cas où nous ne possédons pas d'alimentation stabilisée, il est possible d'alimenter le montage avec une pile 9 volt suivi par un régulateur 5 volt du type 7805 (Figure II-1). Le montage peut être alimenté avec une tension allant de 8 volts à 18 volts. Pour bien protéger le montage contre les inversions de polarité de l'alimentation on ajoute une diode de type 1N4001 en série avec pile 9 volts. Les condensateurs permettent de filtrer l'alimentation 5 volt.

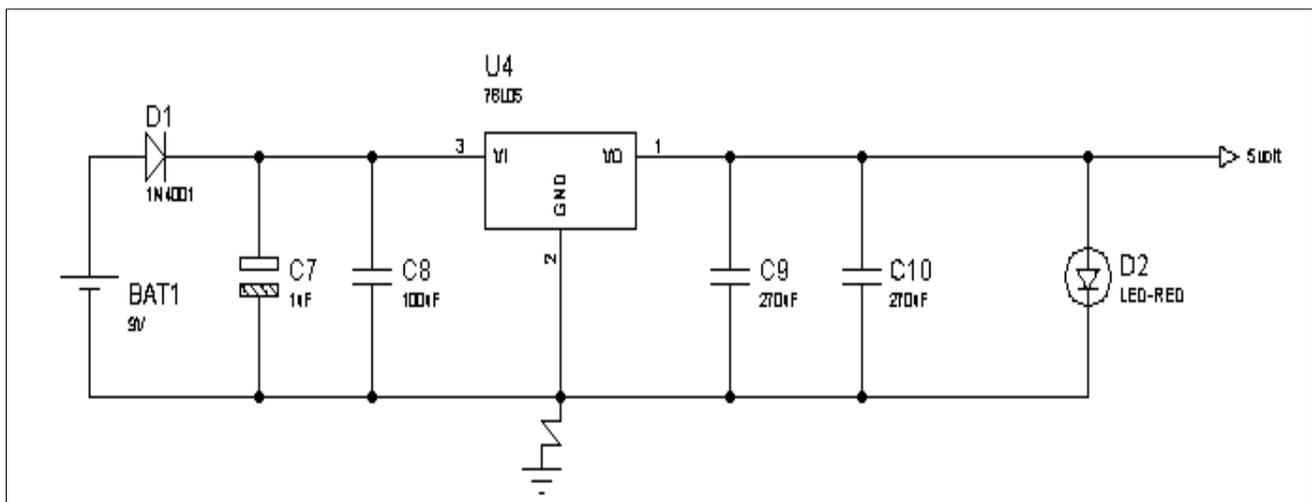


Figure II-1 : Alimentation 9V suivie d'un régulateur 5V (7805).

3. 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. Dans la jungle des références le PIC 16f628 est un des modèles (un des plus petits) de la famille Mid-Range des circuits microcontrôleurs de Microchip. Il possède de nombreuses performances et caractéristiques.

En fait les causes principales du choix du 16f628 sont :

- Une programmation facile en langage C.
- Son faible coût et son petit taille.
- Possède en plus des instructions très puissantes donc un programme à développer réduit.

4. PIC 16F628

4.1 Présentation

Ce composant intègre un microcontrôleur 8 bit, c'est-à-dire un processeur et des périphériques, dans un boîtier DIL (Dual In Line) de 18 broches. Il est réalisé en technologie CMOS et peut cadencer par une horloge allant de 0 à 20 MHz, il doit être alimenté par une tension allant de 3 à 5,5 volts. Les broches du composant (Figure II-2) possèdent plusieurs affectations entre les ports d'E/S, les périphériques et les fonctions système^[7].

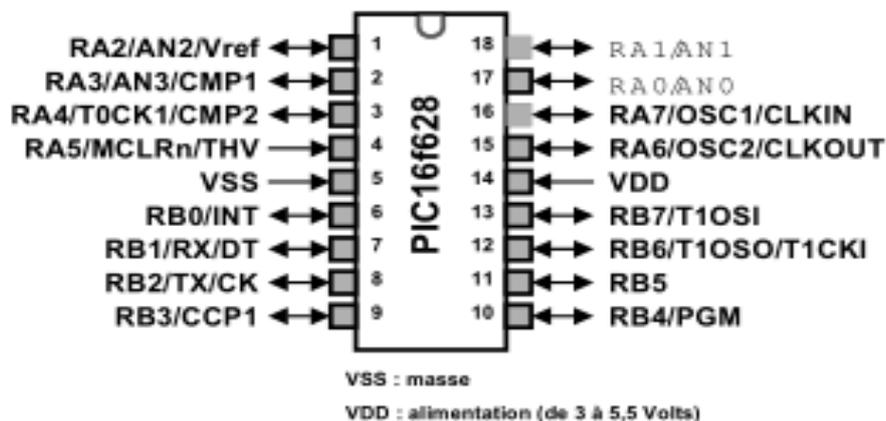
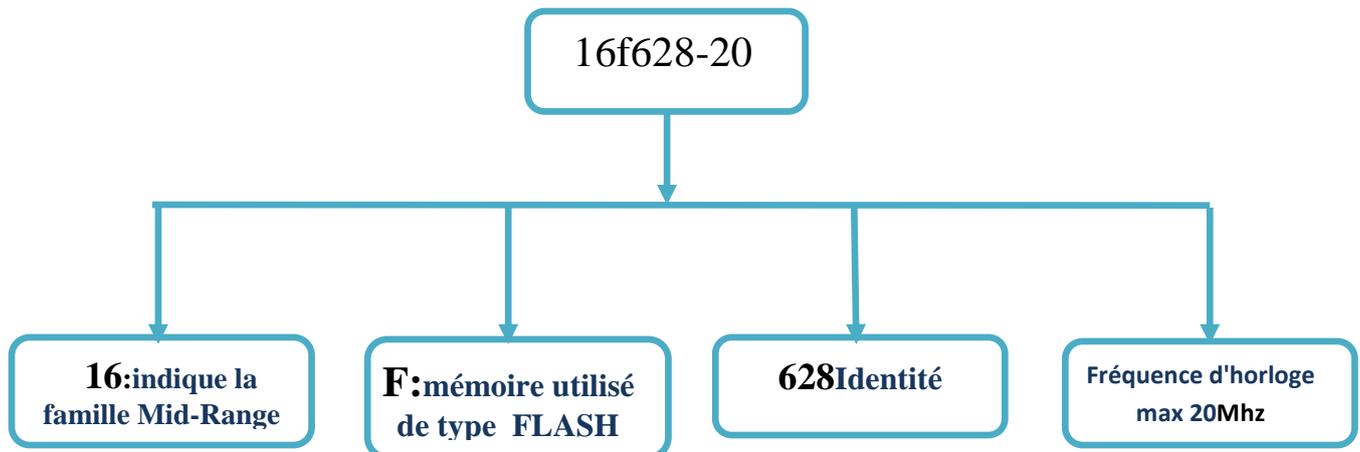


Figure II-2 Brochage du pic 16F628

Voici, les caractéristiques générales du PIC 16F628 fourni par Microchip :

- Mémoire Flash Programme : 2048 instructions.
- Mémoire RAM Données: 224 Octets.
- Mémoire EEPROM Données : 128 Octets.
- Ports d'E/S: 2ports 8bits.
- 3 Timers (8 et 16 bits).
- 1 module Capture/Compare/PWM.
- 2 comparateurs analogiques.
- 1 référence de tension.
- 1 USART (émission/réception série synchrone et asynchrone).

Le diagramme suivant montre l'identification de notre pic 16f628 utilisé :



4.2 Architecture

Jusqu'à une certaine époque, les microcontrôleurs respectaient l'architecture Von Neumann, inventeur de l'ENIAC.

Pendant, celle-ci présente des inconvénients. En effet, la vitesse d'exécution est limitée, les instructions et les données transitaient par le même bus. Pour résumer, son principal défaut était le fait qu'elle ne possédait qu'un bus pour, simultanément, la mémoire programme et la mémoire donnée.

D'où l'architecture Harvard, utilisée maintenant par 16f628 (figure II-3). Sa particularité tient dans le fait qu'il y a deux mémoires accessibles en même temps par le processeur, par l'intermédiaire de deux bus spécifiques. L'un sert pour les données, et l'autre pour les instructions. De ce fait, les deux peuvent être accessibles en même temps, d'où un gain de vitesse, au niveau exécution ^[7].

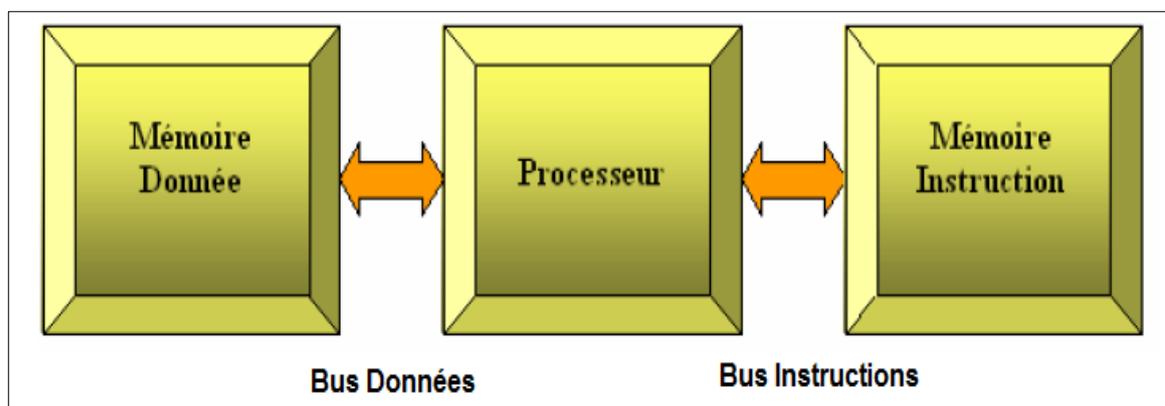


Figure II-3 Architecture Harvard.

4.2.1 Schéma fonctionnel

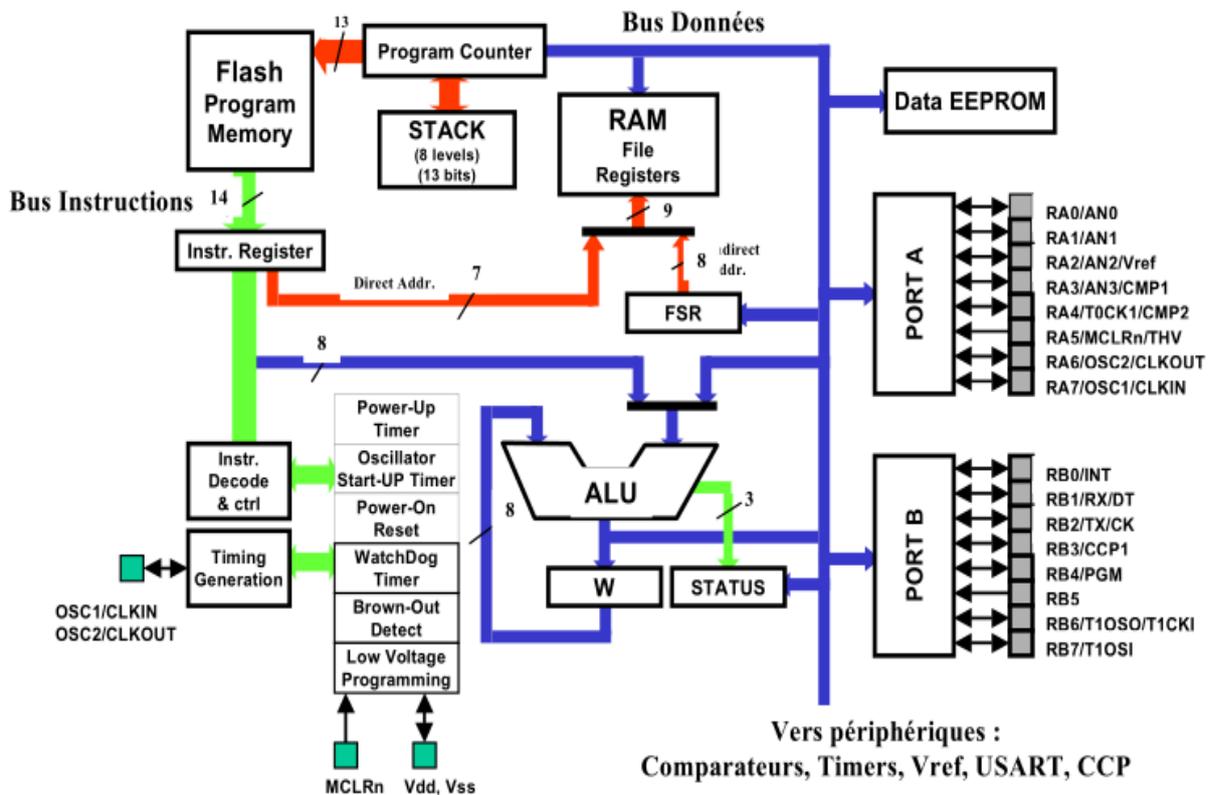


Figure II-4 Schéma fonctionnel du PIC 16f628.

4.2.2 Unité centrale

Le cœur du processeur est composé de 3 entités:

- L'ALU (opération arithmétiques et logique sur 8 bits) et son registre d'état (STATUS).
- Le registre de travail, W (working Register) ou accumulateur.
- Le registre d'usage général (File Registers).

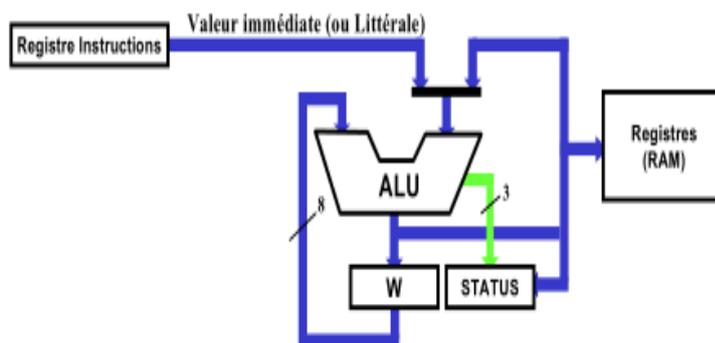


Figure II-5 Cœur d'un processeur PIC.

4.2.3 Mémoire

La mémoire du 628 est divisée en 3 parties comme illustré dans la figure II-4.

- Mémoire programme :

Les instructions du programme sont stockées sur 14 bits dans une zone mémoire dont l'adresse s'étend de 000h à 7FFh, 2048 lignes de programme. Il y a 2 adresses réservées pour les vecteurs d'initialisation (Reset et interruptions) le processeur possède une pile de 8 niveaux pour gérer les adresses de retour de sous programmes ; il n'y a aucun contrôle sur l'état de la pile par le processeur^[8].

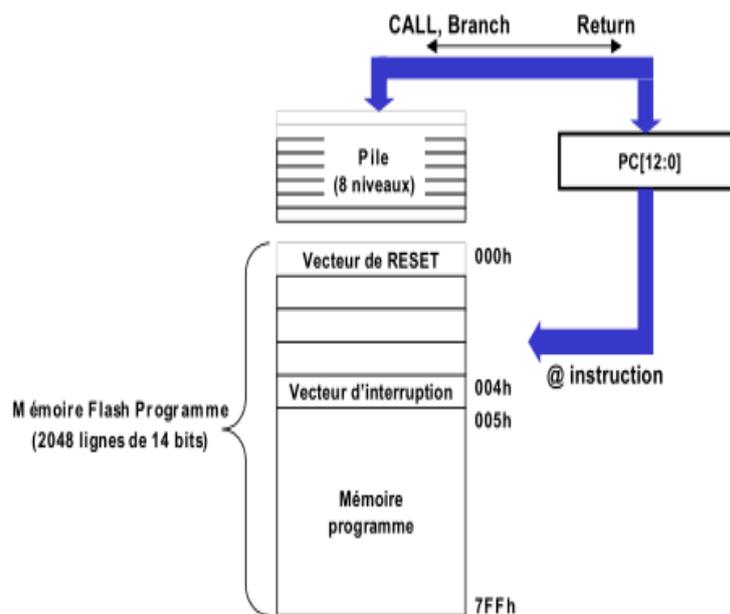


Figure II-6 Mémoire programme.

- Mémoire données (Registres) :

La mémoire donnée correspond aux registres (File Registers) vus par l'unité centrale ; ces registres sont de 2 types :

- Registres « Système » : ils permettent la configuration et la surveillance de l'état du processeur et de ses périphériques ; ces registres sont accessibles en lecture ou en écriture suivant leur fonction ; il y a 34 registres de ce type pour le 16f628.
- Registres d'usage général : ils permettent le stockage de variables; ils sont accessibles en lecture et écriture par le programme ; il y a 224 registres^[8].

- **Mémoire EEPROM :**

Est une zone mémoire non-volatile (EEPROM) de 128 Octets, cette mémoire n'est accessible que par adressage indirect via un registre propre au module.

Il faut savoir qu'une mémoire EEPROM offre un fonctionnement dissymétrique :

- **Lecture :** le fonctionnement est comparable à celui d'une RAM.
- **Ecriture :** l'opération demande un certain temps avant d'être réalisée (mise en route d'une circuiterie d'élévation de tension et effacement de la cellule) dépendant des conditions (**alimentation, température, technologie**); les composants présentent un vieillissement à chaque écriture qui se traduit par un ralentissement de l'écriture.
- **Effacement :** il permet de placer toutes les cellules mémoire à 1.

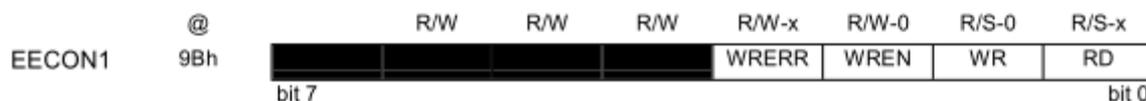
Pour limiter l'usure de l'EEPROM, il peut être intéressant d'analyser le contenu de la cellule mémoire pour savoir s'il n'y a que des 0 à inscrire (pas besoin d'effacer la cellule dans ce cas-là).

Il y a 4 registres pour régir le module :

- **EECON1 et EECON2 :** registres de configuration.
- **EEADR :** registre (7 bits) pointant sur l'adresse concernée.
- **EEDATA :** registre d'indirection.

Le registre EECON1 ne contient que 4 bits pour le contrôle des opérations.

- **WRERR (EEPROM Error Flag):** drapeau indiquant une erreur d'écriture.



- **WREN:** autorisation d'écriture en mémoire.
- **WR:** (ne peut qu'être forcé à 1) Ordre d'écriture, réinitialisé automatiquement en fin d'opération (durée de plusieurs cycles machines suivant conditions).
- **RD:** (ne peut qu'être forcé à 1) Ordre de lecture, réinitialisé automatiquement en fin d'opération (durée 1 cycle machine).

La manipulation de la mémoire EEPROM ne doit se faire que suivant un protocole imposé. La lecture est relativement facile à mettre en œuvre, elle se fait par adressage indirect et lancement d'un ordre de lecture ; l'opération se faisant en 1 cycle, il n'y a pas à se soucier de la disponibilité de la mémoire ^[8].

4.2.4 Ports d'E/S ^[6]

Le 628 dispose de 2 ports d'E/S : A et B ; ceux-ci peuvent servir d'E/S numériques standards ou d'E/S de périphériques internes.

Ces ports sont bidirectionnels, leur configuration se fait par des registres spécifiques « TRISx » ; par exemple : TRISA configure le Port A : On ne peut affecter que deux valeurs différentes de configuration à chaque patte : un '1' pour la mettre en entrée, ou un '0' pour une sortie.

Description des ports :

- **Port A:**

- A0 (RA0/AN0) : entrée pour les comparateurs analogiques.
- A1 (RA1/AN1) : entrée pour les comparateurs analogiques.
- A2 (RA2/AN2/Vref) : entrée pour les comparateurs analogiques et sortie de la tension de référence interne.
- A3 (RA3/AN3) : E/S pour les comparateurs analogiques.
- A4 (RA4/T0CKI) : E/S « Drain Ouvert, entrée d'horloge pour le Timer 0.
- A5 (RA5/MCLRn/THV) : broche de «Reset» et de « programmation haute tension ».
- A6 (RA6/OSC2/CLKOUT) : broche de sortie pour le signal d'horloge.
- A7 (RA7/OSC1/CLKIN) : broche d'entrée pour le signal l'horloge.

Les registres associés au port A sont : PORTA, TRISA, CMCON et VRCON.

- **Port B :**

- B0 (RB0/INT) entrée d'interruption externe.
- B1 (RB1/RX/DT) entrée de l'USART.
- B2 (RB2/TX/CK) sortie ou horloge de l'USART.
- B3 (RB3/CCP1) E/S du module Capture/Compare/PWM.
- B4 (RB4/PGM) contrôle du mode de programmation « basse-tension »
- B5 (RB5) Port bidirectionnel E / S. Interrompez-sur-broche changement. Peut-être logiciel programmé pour faible pull-up interne.
- B6 (RB6/T1OSO/T1CKI) E/S du Timer1 et ligne d'horloge de la programmation du composant.
- B7 (RB7/T1OSI) entrée d'horloge pour le Timer1.

Les registres associés au port B sont : PORTB, TRISB, OPTION et VRCON.

Note : les broches sont configurables en E/S numériques bi- directionnelles la liste au-dessus ne présente alors que les configurations spécifiques à chaque broche.

4.3 Les module

4.3.1 Les Timers

Le premier Timer 0 est un compteur 8 bits (TMR0), accessible en lecture et en écriture, dont la retenue alimente un drapeau d'interruption ; l'horloge de ce compteur peut être interne ou externe par l'intermédiaire d'un pré diviseur (3 bits).

Les registres associés au Timer 0 sont : TMR0, INTCON, OPTION, TRISA.

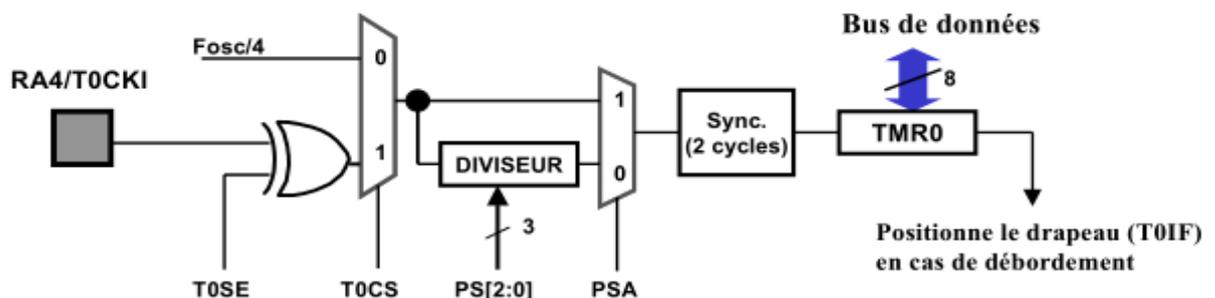


Figure II-7 Timer 0.

Le second Timer1 est sur 16 bits, permettant ainsi d'étendre les possibilités.

Le troisième Timer 2, lui sur 8 bits, possède un pré et un post-diviseur, permettant ainsi de générer un signal PWM^[8].

4.3.2 Comparateurs analogiques^[7]

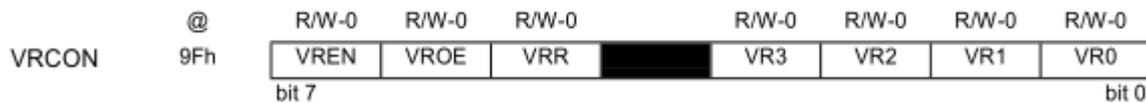
Le module est composé de 2 comparateurs analogiques, alimentés par le port A et la tension de référence interne. Le registre CMCON qui sert à configurer les comparateurs :

CMCON	@ 1Fh	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
		C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
		bit 7							bit 0

- CxOUT: sorties des comparateurs (peuvent être placées sur le Port A)
- CxINV: polarité des sorties
- CIS : multiplexage des entrées : RA0/RA3 et RA1/RA2
- CM [2 :0] : choix du mode (voir annexe).

4.3.3 Tension de référence [7]

Le module tension de référence est constitué d'un réseau de 16 résistances permettant d'alimenter les comparateurs avec une tension de référence.



- VREN (VREF Enable) : Validation du module.
- VROE (VREF Output Enable) : sortie sur RA2.
- VRR (VREF Range Selection) : choix de la plage.
- VR[3 :0] : choix de la valeur (dans la plage).
- Si VRR = 1 : $VREF = (VR[3 :0]/24)*Vdd$.
- Si VRR = 0 : $VREF = Vdd/4 (VR[3 :0]/32)*Vdd$.

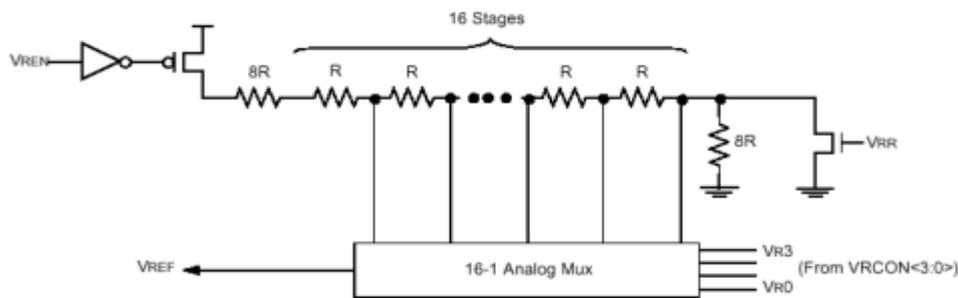


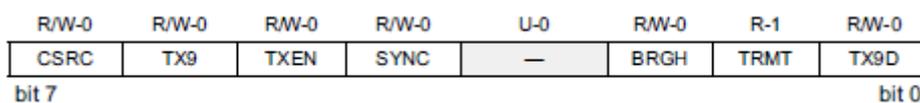
Figure II-8 Module de Tension de référence.

4.3.4 Module USART [7]

USART (Universal Synchronous Asynchronous Receiver Transmitter) est un module de communication série synchrone ou asynchrone. Il est également connu sous le nom de Interface de communication série ou SCI. L'unité USART peut être configurée comme un système asynchrone en duplex intégral qui peut communiquer avec des périphériques tels que des terminaux et des ordinateurs personnels, ou il peut être configuré comme un système synchrone semi-duplex qui peut communiquer avec des périphériques tels que des circuits intégrés, EEPROM série, etc. Les broches RB1 et RB2 ont affectées à l'USART, si celui-ci est activé.

L'unité USART peut être configurée par les deux registres suivants:

1. Registre TXSTA : registre qui contrôle la transmission.



- bit 7 CSRC: bit qui sélectionne la source d'horloge, 1 = mode Master (horloge généré en interne à partir de BRG), 0 = mode esclave (horloge de source externe)
- bit 6 TX9: 1 = Sélectionne la transmission 9-bit, 0 = Sélectionne la transmission 8-bit
- bit 5 TXEN: 1 = Transmission activée, 0 = Transmit désactivé
- bit 4 SYNC: sélectionne le mode USART ; 1 = mode synchrone, mode = 0 Asynchronous
- bit 2 BRGH: bit utilisation en mode asynchrone ; 1 = grande vitesse, 0 = faible vitesse
- bit 1 TRMT: bit indique l'état de registre de transmission ; 1 = TSR vide, 0 = TSR complet
- bit 0 TX9D: 9ème bit de données d'émission. Peut-être bit de PARITÉ.

2. Registre RCSTA : registre qui contrôle l'état de la transmission

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D
bit 7							bit 0

- bit 7 SPEN: active la communication série (RB1/ RX et RB2/TX doivent configurer en entrées).
- bit 6 RX9: 1 = Sélectionne la réception de 9 bits, 0 = Sélectionne la réception 8-bit
- bit 5 SREN: Le mode synchrone - maître: 1 = Active réception seule, 0 = Désactive seule recevoir. Ce bit est effacé après la réception est terminée.
- bit 4 CREN: 1 = recevoir en continue, 0 = désactive de recevoir en continue.
- bit 3 ADEN: le mode asynchrone de 8 bits (RX9 = 0): Non utilisé dans ce mode
- bit 2 FERR: bit d'erreur de cadrage 1 = erreur de cadrage, 0 = Pas d'erreur de cadrage
- bit 1 OERR: bit d'erreur de dépassement ; 1 = erreur (peut être effacé en effaçant peu CREN), 0 = Pas d'erreur de dépassement
- bit 0 RX9D: 9 bits de données reçues (Peut être bit de parité).

4.4 Configuration de l'horloge

Le composant dispose de 8 modes de configuration de l'horloge :

- **ER (External Resistor)** : l'horloge est réalisée par un oscillateur de type RC, la capacité est interne au composant et la résistance, externe, permet de fixer la fréquence désirée ; ce mode existe en 2 variantes suivant que l'on utilise la broche RA6 en E/S ou en sortie d'horloge (CLKOUT).

- **INTRC (Internal RC)** : même principe que précédemment mais avec une résistance interne.
- **EC** : on doit fournir un signal d'horloge au composant (broche RA7).
- **HS, XT, LP**: modes « classiques » pour un microcontrôleur correspondant à placer un quartz à un inverseur interne au composant (figure suivante) ; le choix du mode dépend de la fréquence de l'horloge ^[8].

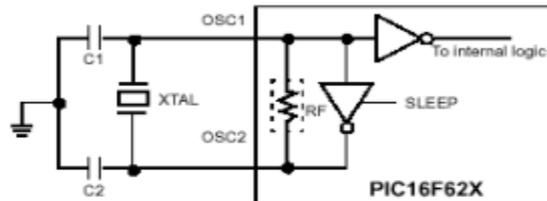


Figure II-9 configuration d'oscillateur HS, XT et LP.

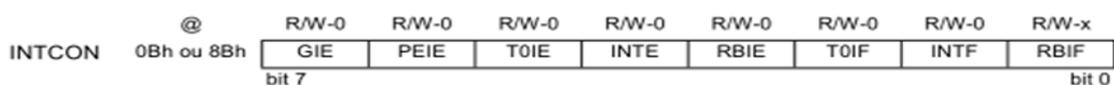
4.5 Interruptions

Une interruption est une action extérieure à un programme obligeant celui-ci à traiter un sous-programme prioritaire logé à une adresse bien spécifique de la mémoire du microcontrôleur. Sur certains microcontrôleurs il existe une hiérarchisation des interruptions on parlera dans ce cas de niveau d'interruption avec gestion des priorités selon la nature de l'évènement qui survient ^[6].

Le processeur du PIC 16f628 dispose de 10 sources d'interruption :

- Externe : à partir de la broche RB0/INT.
- Débordement du Timer 0.
- Changement de l'état des broches RB[7:4].
- Module comparateurs analogiques.
- USART.
- Module CCP (Capture/Compare/PWM).
- Débordement du Timer 1.
- Reconnaissance d'occurrence (Match) du Timer 2 ^[8].

Pour que le PIC 16f628 accepte une ou plusieurs de ces sources d'interruptions il faut lui préciser par l'intermédiaire de bits que l'on déclare dans un registre nommé INTCON (INTerruptCONtroll).



Le nom des bits est significatif, en effet les noms se terminant par F sont des flags, ces bits sont positionnés lors de l'interruption et permettent à l'utilisateur de savoir quel évènement à l'origine de l'interruption, il est impératif que l'utilisateur repositionne ce flag à 0. Le nom des bits se terminant par E sont des bits de validation d'une interruption.

Les fonctions des bits sont les suivants :

- GIE : (GlobalInterruptEnable) : toutes les interruptions.
- PEIE : (Peripheral Int. Enable) : interruptions des périphériques.
- TOIE : (timer 0 Int. Enable) : interrupt.débordement du Timer 0.
- INTE : (External Int. Enable) : interruption de RB0/INT.
- RBIE : (Port B Change Int. Enable) : interruption de RB7-RB4.
- TOIF : flag d'IT du Timer 0.
- INTF : flag d'IT du RB0/INT.
- RBIF : flag d'IT du changement de RB7-RB4. [6]

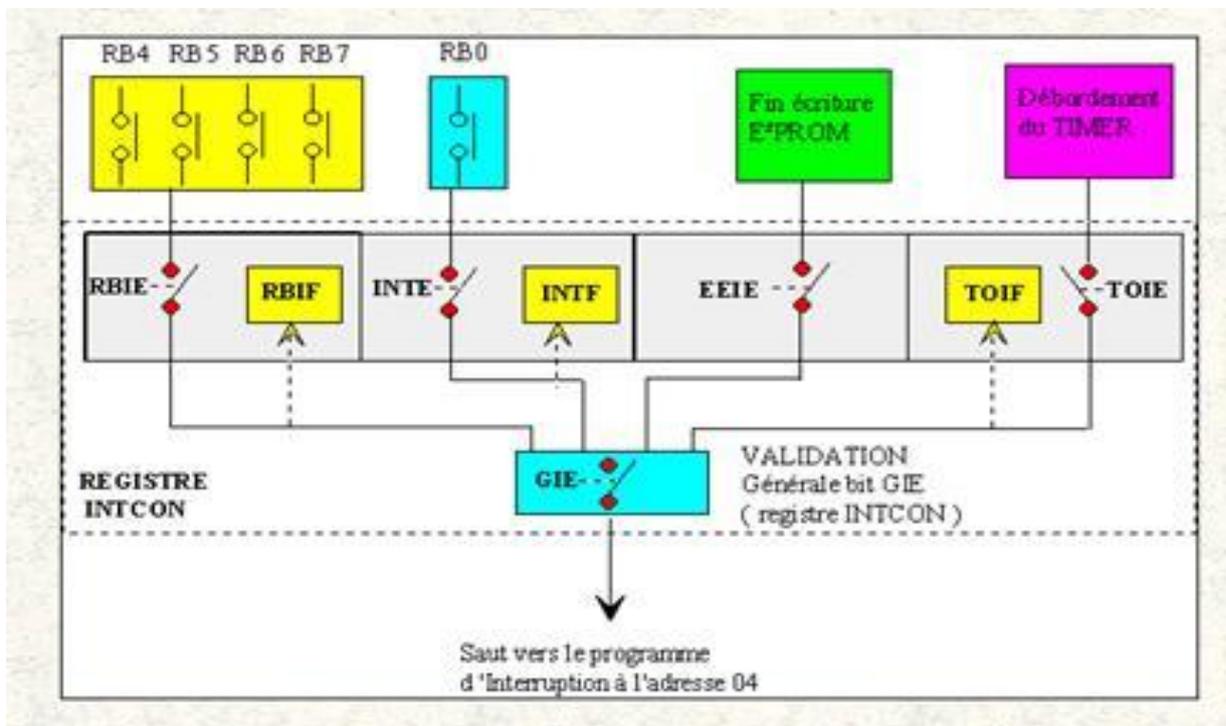


Figure II-10 Synoptique général des interruptions sur le 16F628.

Les autres sources d'IT sont gérées et réalisées dans les registres de contrôle tel que PIE1, PIR1, CMCON.

4.6 Watchdog

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

4.7 Le reset

Le reset sur le 628 peut être de deux types. Classique manuel, ou bien automatique Watchdog. Le cas de réaliser un reset automatique en cas de baisse anormale de la tension. Cette fonction de surveillance est activée par le positionnement à 1 d'un fusible nommé BODEN^[6].

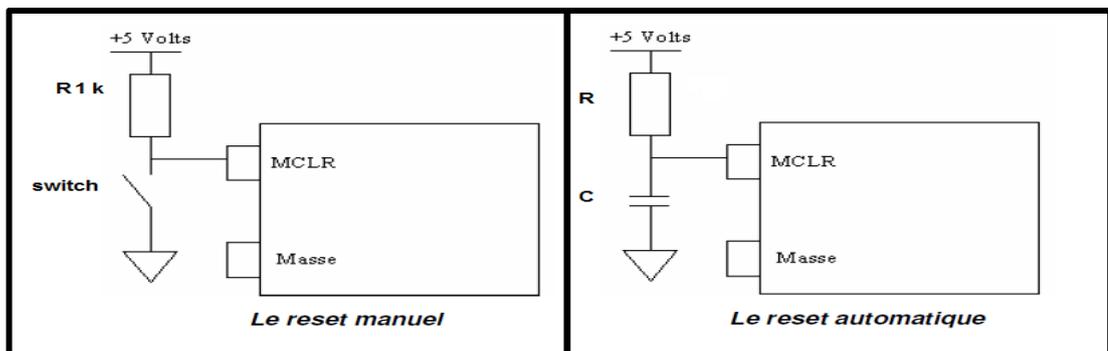


Figure II-11 les modes du reset.

5. Les afficheurs 7 segment^[9]

Les afficheurs 7 segments sont des afficheurs numériques composés de 7 LED a, b, c, d, e, f et g que l'on allume ou non pour afficher un nombre décimal de 0 à 9.

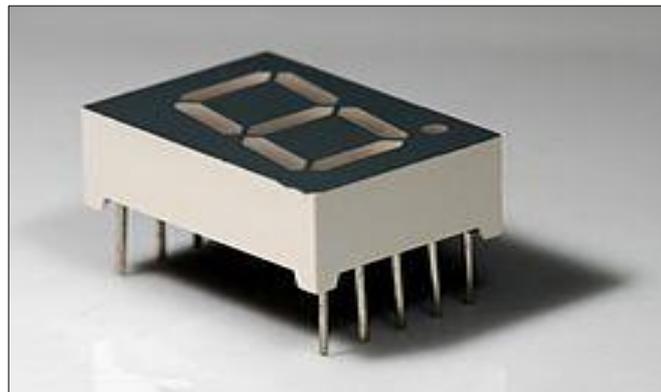


Figure II-12 Afficheur 7 segments.

Il existe deux types d'afficheurs 7 segments :

5.1 Afficheurs à anode commune

Toutes les anodes sont reliées et connectées au potentiel haut. La commande du segment se fait par sa cathode mise au potentiel bas.

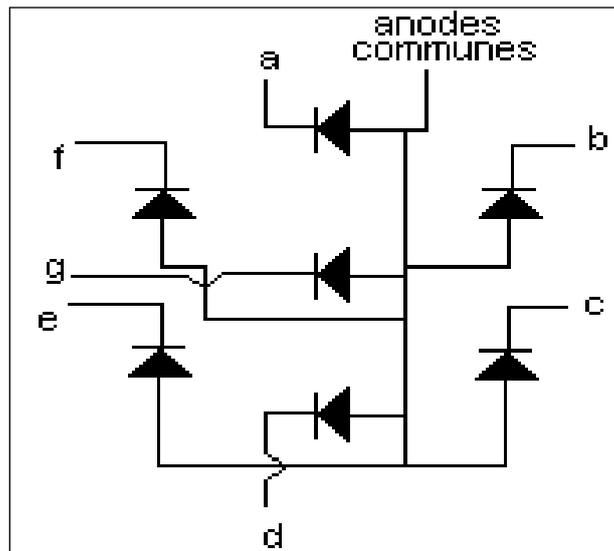


Figure II-13 Schéma interne d'un afficheur à anode commune.

5.2 Afficheurs à cathode commune

Toutes les cathodes sont reliées et connectées au potentiel bas. La commande du segment se fait par son anode mise au potentiel haut.

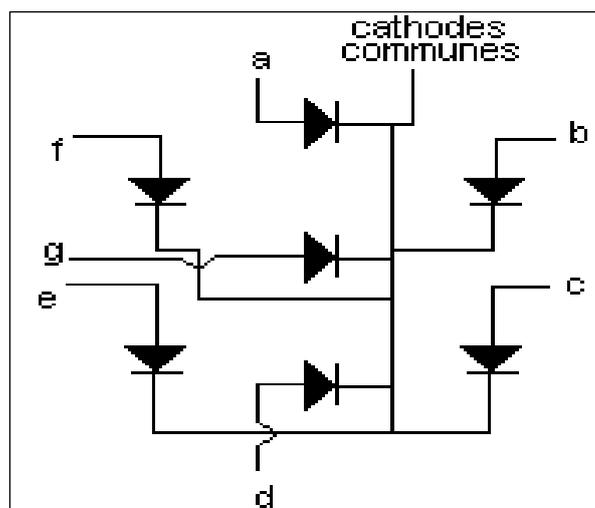


Figure II-14 Schéma interne d'un afficheur à cathode commune.

6. Les divers détecteurs

Un détecteur est un capteur qui délivre des signaux électriques correspondant au franchissement d'un seuil (niveau, position). L'information est alors en logique binaire 0 ou 1. Un capteur est un appareil destiné à transformer une grandeur physique mesurable (vitesse, pression, lumière, température) en un signal électrique de nature analogique ou numérique.

6.1 Choix des détecteurs

La grande variété des modèles, des technologies permettent un choix très large mais parfois difficile à maîtriser. Toutefois on peut définir des critères de choix qui porteront principalement sur :

- La fonction à réaliser (signal analogique, numérique, logique binaire),
- Le mode de fonctionnement (statique ou dynamique),
- La détection à réaliser : (type de matériaux à détecter, solide, liquide, etc.) contrôle de passage de pièces, déplacement, niveau, vitesse),
- La technologie du système (château d'eau, alarmes anti intrusion et techniques (incendie),
- L'environnement : agents extérieurs corrosifs, humide, poussiéreux...
- Les contraintes mécaniques : vitesse de commutation (comptage rapide, vitesse à détecter très grande, fréquence de manœuvres par heure ...
- Les caractéristiques électriques : tension, nature de la tension, tension d'isolation, intensité, pouvoir de coupure, longueur matière et nombre de fils^[10].

Voici un algorithme de choix des capteurs les plus communs :

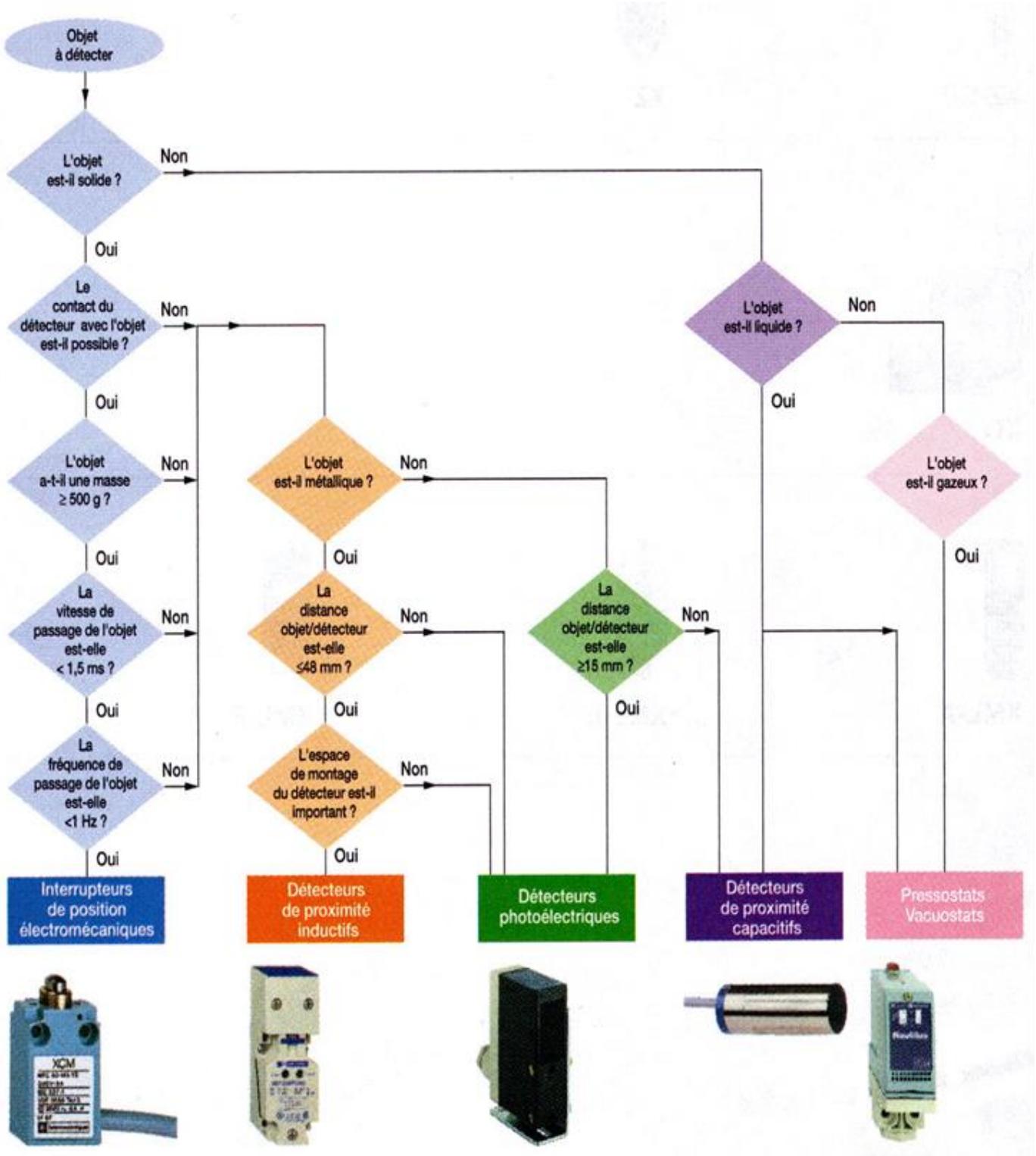


Figure II- 15 L’algorithme de choix des détecteurs

Selon l'algorithme les détecteurs photoélectriques est un bon choix dans une installation basée sur la détection de mouvement.

6.2 Détecteurs photoélectrique

Les détecteurs photoélectriques sont des appareils capables de détecter des objets à des distances très grandes (du cm à plusieurs dizaines de mètres).

Ils sont de deux types :

- Sans réflecteur avec un boîtier émetteur et un boîtier récepteur : barrière optique par transmission.
- Avec ou sans réflecteur mais un seul boîtier contenant émetteur et récepteur type réflexe : barrière optique avec réflecteur ou réflexion^[10].

6.2.1 Barrière optique par transmission

Les barrières optiques sont largement utilisées, tant dans la vie courante (commande de porte automatique, alarme..) que l'industrie (détecteur d'objet sur un tapis roulant...).

- Principe :

Une barrière optique est l'association d'un émetteur et d'un récepteur contrôlant la continuité d'un faisceau lumineux qui peut être interrompu par un obstacle. L'émetteur est donc en général une diode infrarouge ou parfois une diode laser pour une portée plus grande, et le récepteur est plus souvent un phototransistor au silicium. Celui-ci possède une meilleure sensibilité qu'une photodiode ou photorésistance et ne présente pas d'inconvénient dans une application où une grande rapidité n'est pas nécessaire^[9].

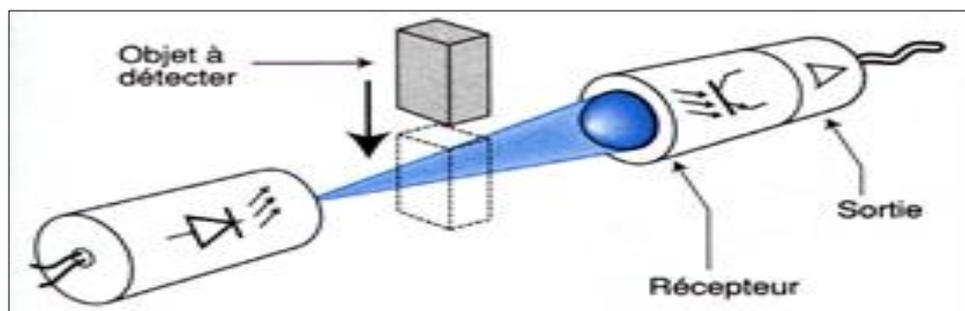


Figure II-16 Barrière par transmission.

6.2.2 Barrière à transmission avec réflecteur ou réflexion

Il peut être commode de rassembler émetteur et récepteur du même côté. On ajoute alors un réflecteur de l'autre côté : c'est dispositif similaire à un catadioptré, qui revoie la lumière. Cette solution est néanmoins à éviter pour détecter des objets à la surface polie, pour lesquels des perturbations seraient à craindre. Dans les barrières à réflexion il est possible de détecter la lumière reçue par réflexion sur l'obstacle, le récepteur ne reçoit pas la lumière de l'émetteur en absence d'obstacle, mais il est éclairé par réflexion sur l'objet à détecter. Dans certains cas, c'est la surface brillante de l'obstacle qui revoie la lumière, mais dans d'autres cas, il faut ajouter une pastille réfléchissante sur l'objet^[9].

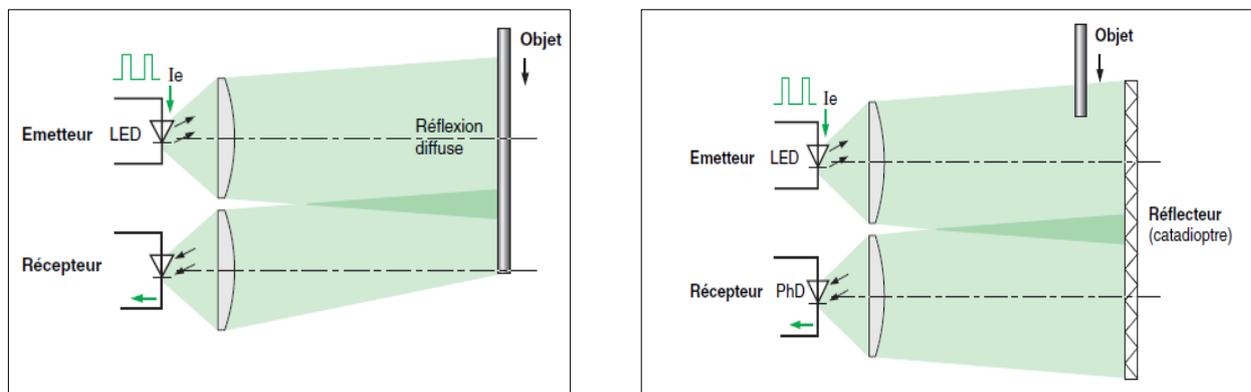


Figure II-17 Barrière à transmission avec réflexion et réflecteur.

7. Conclusion

Pour notre projet, nous avons choisi d'utiliser le PIC pour son coût et sa disponibilité. Pour l'affichage, nous avons choisi afficheur 4 x 7 segments cathode commune pour faciliter ces pilotage. Ainsi de réaliser un détecteur de passage.

Le prochain chapitre sera dédié à la conception du système et le développement des algorithmes.



Chapitre

III

**Conception
Et Programmation**

1. Introduction

Le projet réalisé est un système de comptage de 0 à 9999, basé sur un petit microcontrôleur PIC 16f628. A la présence d'un objet qui coupe le faisceau lumineux d'un capteur optique ce dernier génère une impulsion permettant d'incrémenter notre afficheur. La visualisation du nombre à afficher est assurée par un afficheur de type 4 x 7 segments et dans l'application HyperTerminal à l'aide d'une communication série RS232.

2. Implantation de schémas électriques du système de comptage

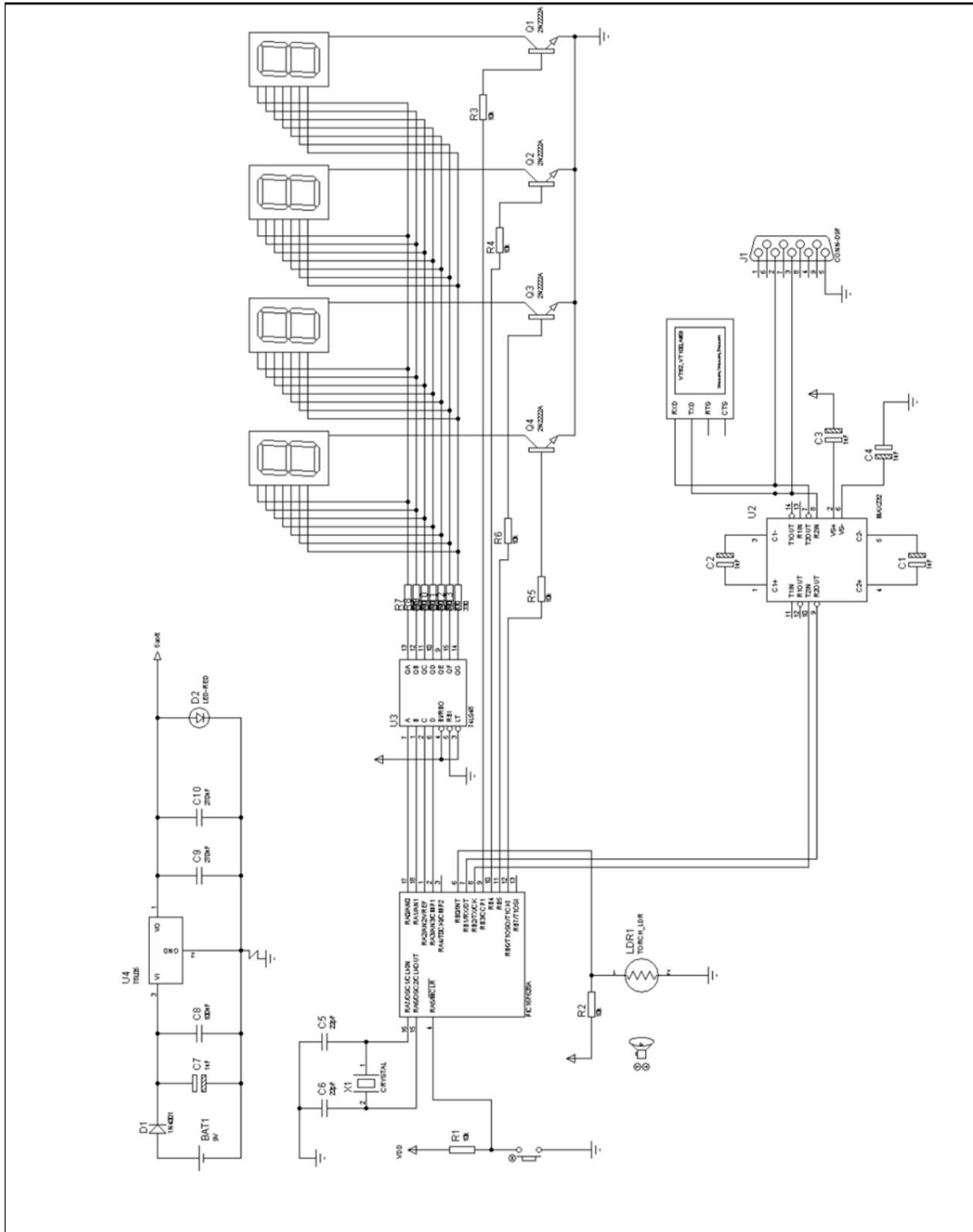


Figure III-1 : Schémas électrique du système.

Voici le schéma du système. Chaque segment des afficheurs cathode commune est relié à la sortie du décodeur BCD de A à G, le PORTA qui doit fournir le code de chaque chiffre. Les transistors sont reliés au PORTB (de RA3 à RA6) et active ou non les digits correspondants, le premier digit (à droite) est branché sur RB3.

3. Commande multiplexé d'afficheurs 4 x 7 segments

Nous avons plus d'un afficheur à utiliser. Le recours au multiplexage est quasiment impératif afin de ne pas monopoliser toutes les lignes d'entrées / sorties de notre PIC.

Le multiplexage consiste à allumer successivement les afficheurs les uns après les autres, très rapidement. On ne verra pas de clignotement grâce à la persistance rétinienne, l'œil a l'impression que tous les afficheurs sont allumés en permanence ^[4].

Le déroulement se fera comme suivant:

- Activation du premier digit > Affichage du chiffre > désactivation du premier digit.
- Activation du deuxième digit > Affichage > désactivation du deuxième digit.
- Activation du troisième digit > Affichage > désactivation du troisième digit.
- Activation du quatrième digit > Affichage > désactivation du quatrième digit.

Les segments des afficheurs sont commandés par les sept lignes du décodeur 4511 BCD à 7-segment. Le circuit accepte 4 bits binaire codé décimal (BCD), en fonction des états des entrées auxiliaires générées par les 4 premières broches du port A, et décode ces données pour afficher les chiffres de 0 à 9 ^[14].

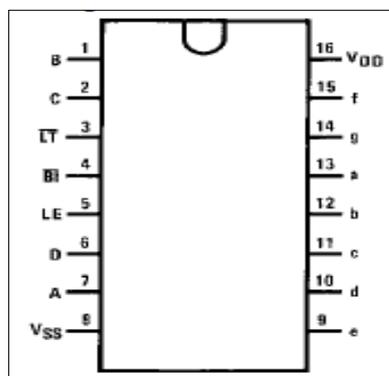
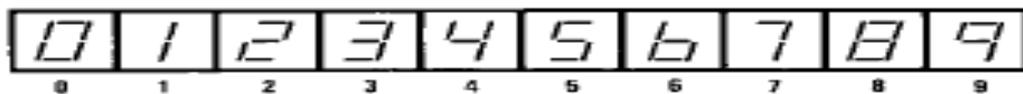


Figure III-2 : Brochage du 4511.

- Table de vérité du décodeur 4511 :

Inputs							Outputs							
LE	BI	LT	D	C	B	A	a	b	c	d	e	f	g	Display
X	X	0	X	X	X	X	1	1	1	1	1	1	1	B
X	0	1	X	X	X	X	0	0	0	0	0	0	0	
0	1	1	0	0	0	0	1	1	1	1	1	1	0	0
0	1	1	0	0	0	1	0	1	1	0	0	0	0	1
0	1	1	0	0	1	0	1	1	0	1	1	0	1	2
0	1	1	0	0	1	1	1	1	1	1	0	0	1	3
0	1	1	0	1	0	0	0	1	1	0	0	1	1	4
0	1	1	0	1	0	1	1	0	1	1	0	1	1	5
0	1	1	0	1	1	0	0	0	1	1	1	1	1	6
0	1	1	0	1	1	1	1	1	1	0	0	0	0	7
0	1	1	1	0	0	0	1	1	1	1	1	1	1	8
0	1	1	1	0	0	1	1	1	1	0	0	1	1	9
0	1	1	1	0	1	0	0	0	0	0	0	0	0	
0	1	1	1	0	1	1	0	0	0	0	0	0	0	
0	1	1	1	1	0	0	0	0	0	0	0	0	0	
0	1	1	1	1	0	1	0	0	0	0	0	0	0	
0	1	1	1	1	1	0	0	0	0	0	0	0	0	
0	1	1	1	1	1	1	0	0	0	0	0	0	0	
1	1	1	X	X	X	X				*				*

- Affichage des chiffres de 0 à 9 :



Tandis que les cathodes communes sont validées à tour de rôle par des transistors de type NPN « 2N2222 » commandés par l'autre port B.

Le 2N2222, également reconnu comme PN2222, est un transistor NPN de faible puissance bipolaire. Il sert à la fois à des applications de commutation que l'amplification. Vous pouvez amplifier des courants faibles à petite ou moyenne tension ; comme il peut fonctionner à des fréquences assez élevées ^[datasheet].



Figure III-3 : le transistor 2N2222.

Dans notre montage le 2N2222 a pour fonction d'interrupteur quand il reçoit un courant dans la base « B » le transistor entre en conduction. La tension présente entre collecteur « C » et la masse est proche de la tension d'alimentation, contrairement la tension est égal à 0 volt.

4. La communication série RS232 entre PIC et PC

Les communications série sont utilisés dans les systèmes à base de microcontrôleurs, principalement en raison du manque des E / S en broches. Pour les communications longue distance, le transfert de données série est plus simple et plus rentable que les connexions matérielles nécessaires à la liaison de données qui peuvent être réduites à trois fils (Tx, Rx et GND). La transmission série de type RS232 est une des transmissions asynchrones car il n'y a pas de signal d'horloge dans le câble ; cela implique que l'émetteur et le récepteur soient configurés au même débit, que l'on exprime en bits par seconde ^[4].

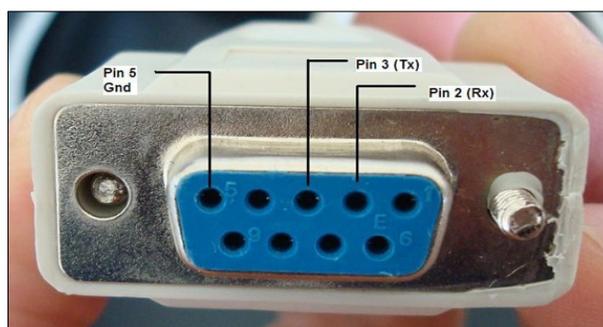


Figure III-4 : broches de signal RS232 dans un connecteur femelle DB-9.

Les paramètres de la liaison RS232 sont :

- Bauds par seconde (vitesse de transmission) ;
- Pas de bit de parité ;

- Au repos, le niveau logique est 1 ;
- Le début de la transmission est marqué par un bit Start au niveau 0, suivi des 8 bits de données ; 1 bit d'arrêt (Stop) ; La fin de la transmission est marquée par un bit Stop (niveau 1).

Le port série du PC (également connu sous le port COM) utilise la norme RS232 pour la communication série. Notez que les niveaux logiques de la liaison RS232 ne sont pas compatibles avec la logique TTL ^[4]. Le rôle d'un circuit intégré de type MAX232 est ce qu'un convertisseur de niveau aura pour effet est de convertir les signaux niveau RS232 (HIGH=-12V LOW=+12V) du PC vers les niveaux TTL (HIGH=+5V LOW=0V) du PIC et le contraire.



Figure III-5 : Rôle du circuit max232.

En utilisant des pompes de charges avec les 4 capacités de 1uf, il génère le +12V et -12V à partir de son alimentation 5Volts ^[14].

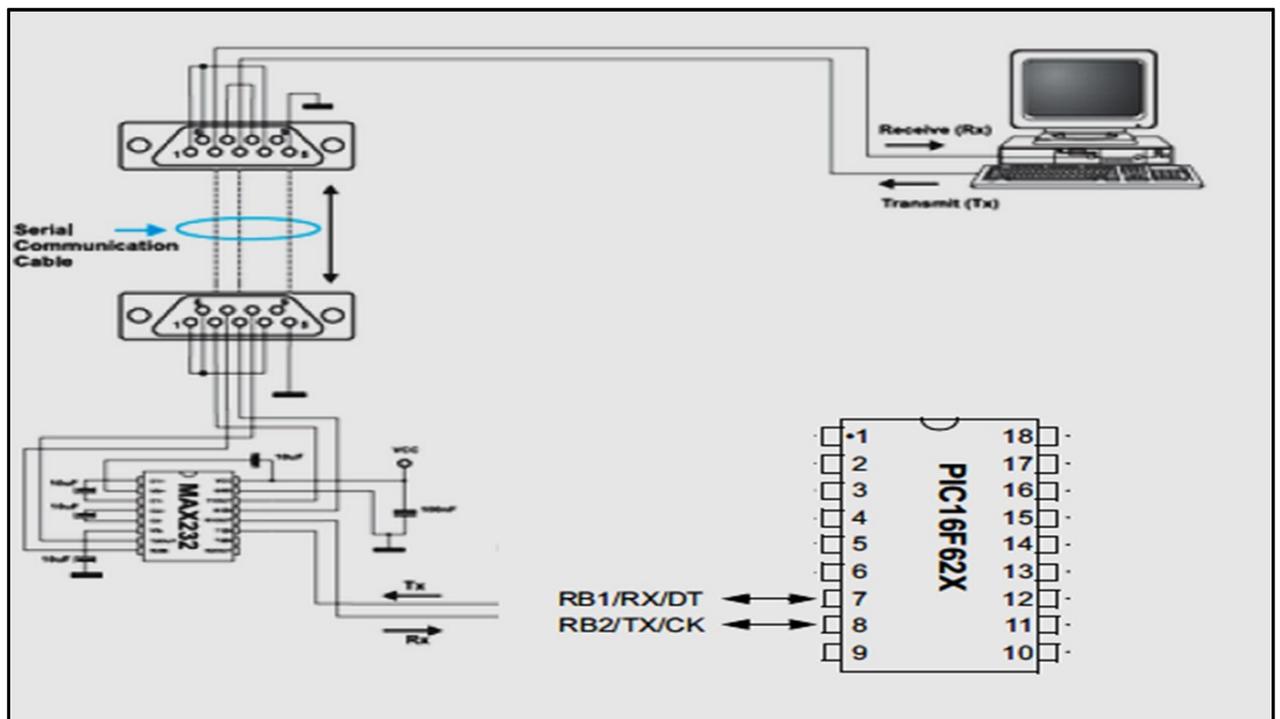


Figure III- 6 (a) : Brochage du circuit max232 entre le PIC et PC.

PIC 16F628 possède un module USART (Universal Synchronous Asynchronous Receiver Transmitter) qui permet de communiquer avec une large gamme de périphériques série tels que, les écrans LCD, les ordinateurs personnels, etc. Le module USART dispose de deux modes de fonctionnement: synchrone (nécessite une horloge synchronisée entre l'émetteur et le récepteur) et asynchrone (pas d'horloge de synchronisation requise).

Le module USART à l'intérieur du microcontrôleur PIC16F628A prend en charge les deux types de communications en série, mais il est mieux adapté pour la méthode asynchrone. En mode asynchrone, RB2 agit comme une transmission de données (TX) de sortie et RB1 en tant que réception de données (RX) d'entrée. Un octet de données série est envoyé sous forme de chaîne de 10 bits; un bit de départ, huit bits de données et un bit d'arrêt, comme illustré ci-dessous ^[7].

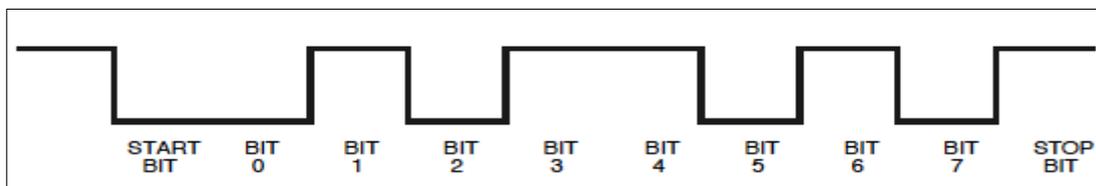


Figure III- 6 (b) : Le format de la trame des données transmises selon la norme RS232.

Pour cela on veut établir un lien de données série dans les deux sens entre le microcontrôleur PIC et un PC à travers une liaison RS-232. Afin d'afficher le nombre de pièces compté dans l'application HyperTerminal.

5. Réalisations de la barrière lumineuse

Le circuit décrit ici permet de détecter la coupure d'un faisceau lumineux de type laser faible puissance (Figure 3).

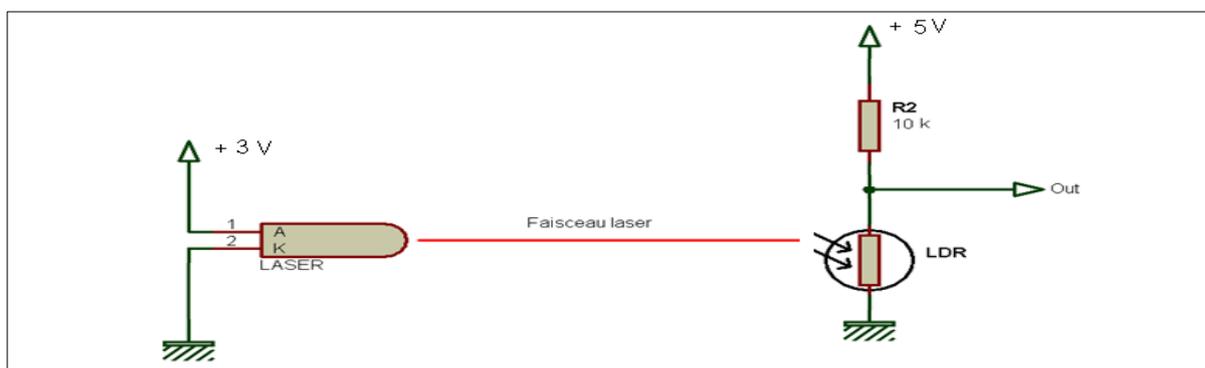


Figure III-7 : Schémas électrique de la barrière lumineuse.

5.1 Emission du faisceau lumineux

Le faisceau lumineux est issu d'un pointeur laser, ce dernier alimenté par une pile 3Volts. L'effet laser (acronyme de light amplified by stimulated emission of radiation, amplificateur de lumière par émission stimulée de rayonnement) permet de réaliser des sources de lumière aux propriétés intéressantes.

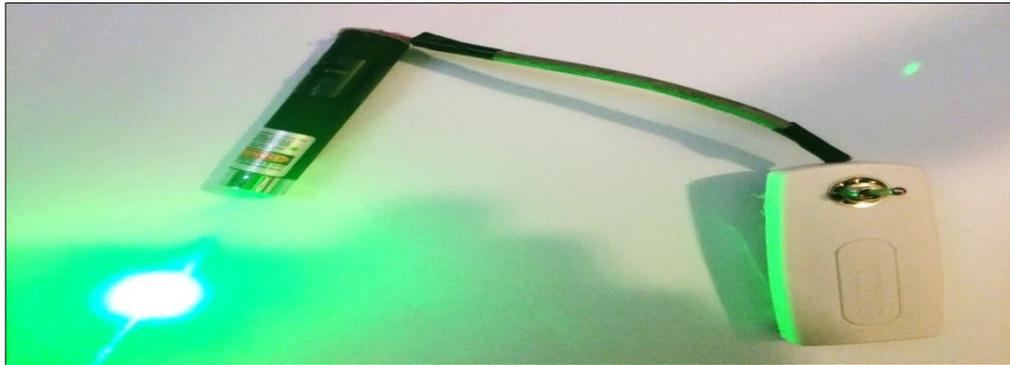


Figure III-8 : Diode laser.

5.2 Détection du faisceau lumineux

La détection du faisceau lumineux se fait très simplement par un composant appelé LDR (light dependent resistor) ou photorésistance.

En appliquant une tension continue de 5 volts à un diviseur résistif formé d'une résistance $R=10\text{ K}\Omega$ et la photorésistance aux bornes de laquelle on prélève la tension (Figure 3).

Une LDR est un dipôle semi-conducteur sans jonction, dont la valeur résistive dépend de la quantité de lumière qu'il reçoit ^[9].

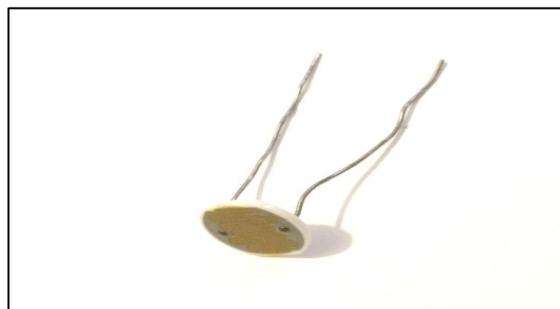


Figure III-9 : Photorésistance

Ici avec un faisceau laser, la résistance est très basse, de l'ordre de 100 ohms. Quand le faisceau est coupé, la résistance peut atteindre plus de 200KΩ dépend de la lumière ambiante. Pour limiter l'influence de la lumière ambiante, il convient de placer la LDR au fond d'un tube en plastique noir. Quand la LDR est frappée par le faisceau laser, sa résistance est très faible. Et la tension prélevé égal environ 0.05 V (on applique diviseur de tension pour calculer la tension Out) :

$$\text{Out} = V \times \left(\frac{\text{LDR}}{R + \text{LDR}} \right) \quad \Rightarrow \quad \text{Out} = 5 \times \left(\frac{0.1}{0.1 + 10} \right) \quad \Rightarrow \quad \text{Out} = 0.049 \text{ volts.}$$

Si le faisceau lumineux est coupé, la LDR se trouve dans l'ombre, sa résistance très élevée. Ce qui fait la tension Out soit environ 5V.

$$\text{Out} = V \times \left(\frac{R}{R + \text{LDR}} \right) \quad \Rightarrow \quad \text{Out} = 5 \times \left(\frac{200}{210} \right) \quad \Rightarrow \quad \text{Out} = 4.76 \text{ volts.}$$

Nous disposons donc sur la sortie « Out » d'un état logique bas au repos et d'un front montant quand le faisceau est coupé.

La sortie Out qui doit être branché avec le pic 16f628 pour délivrer l'information du passage.

6. Partie Logiciel

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 de nouveaux compilateurs qui permettent de programmer avec les langages haut niveau telsque MPLAB, mikroC, Flowcode...

Ces environnements comportent aussi des bibliothèques qui permettent de faciliter le développement ^[11].

Il existe plusieurs outils de développement. Dans notre recherche de l'outil que nous allons utiliser pour programmer notre PIC, nous avons optés pour le langage C. Ainsi d'utiliser l'environnement de développement MikroC PRO for PIC de Mikroelektronika.

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

6.1 Langage et compilateur mikroC for PIC^[12]

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

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

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

Une description détaillée de toutes les options disponibles dans ce compilateur est très long. Au lieu de cela, nous allons décrire le processus d'écriture d'un programme en langage mikroC.

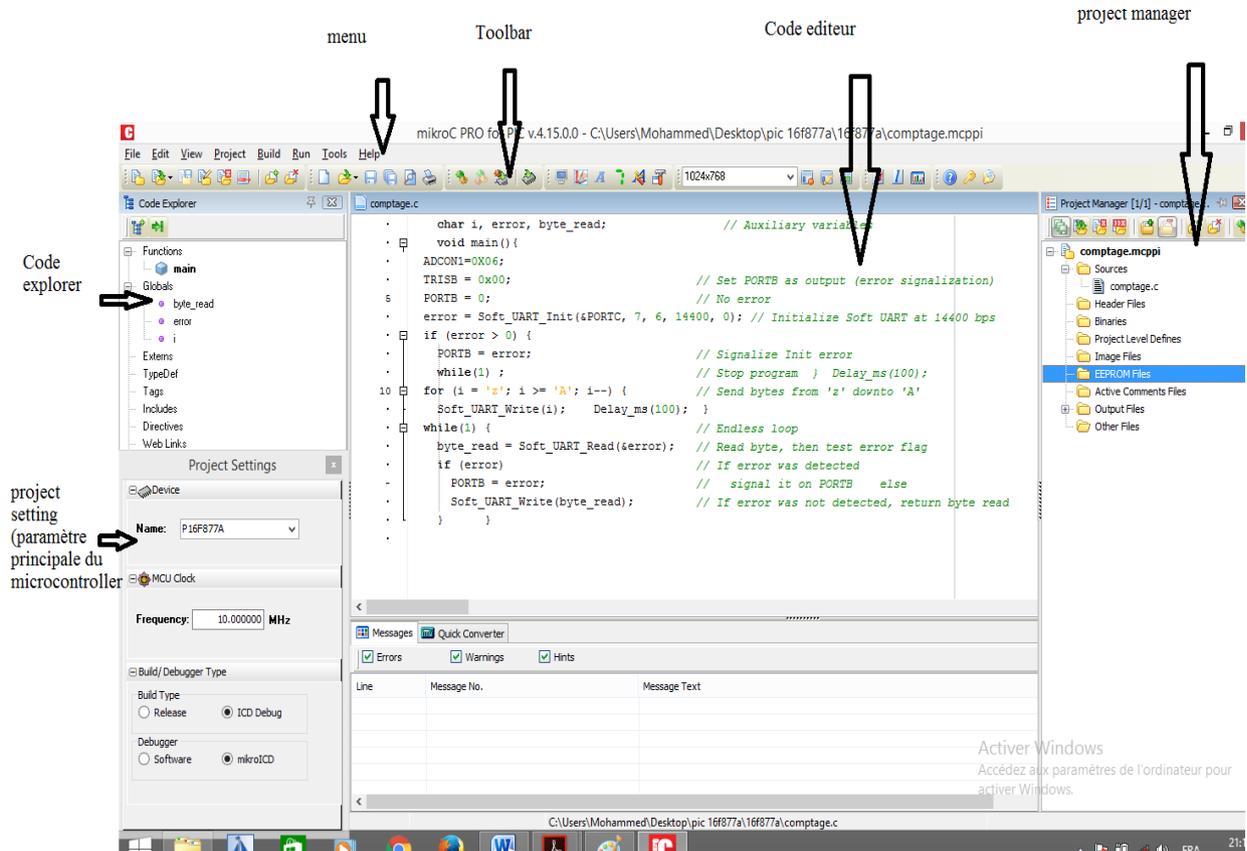
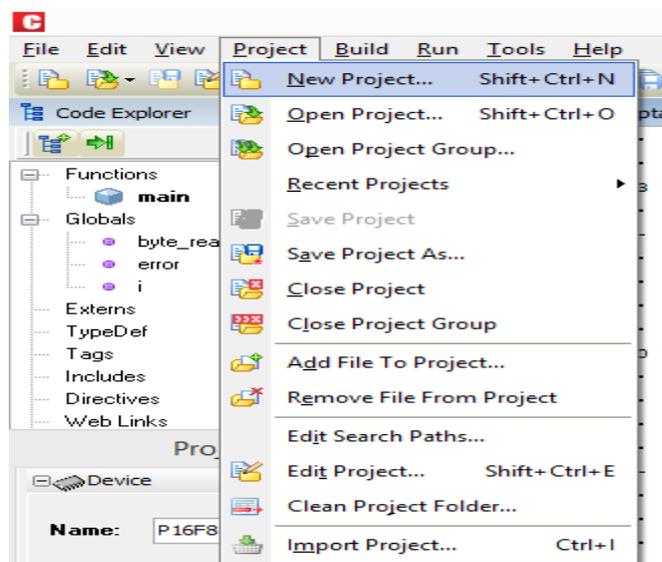


Figure III-10 : L'environnement IDE du compilateur mikroC PRO.

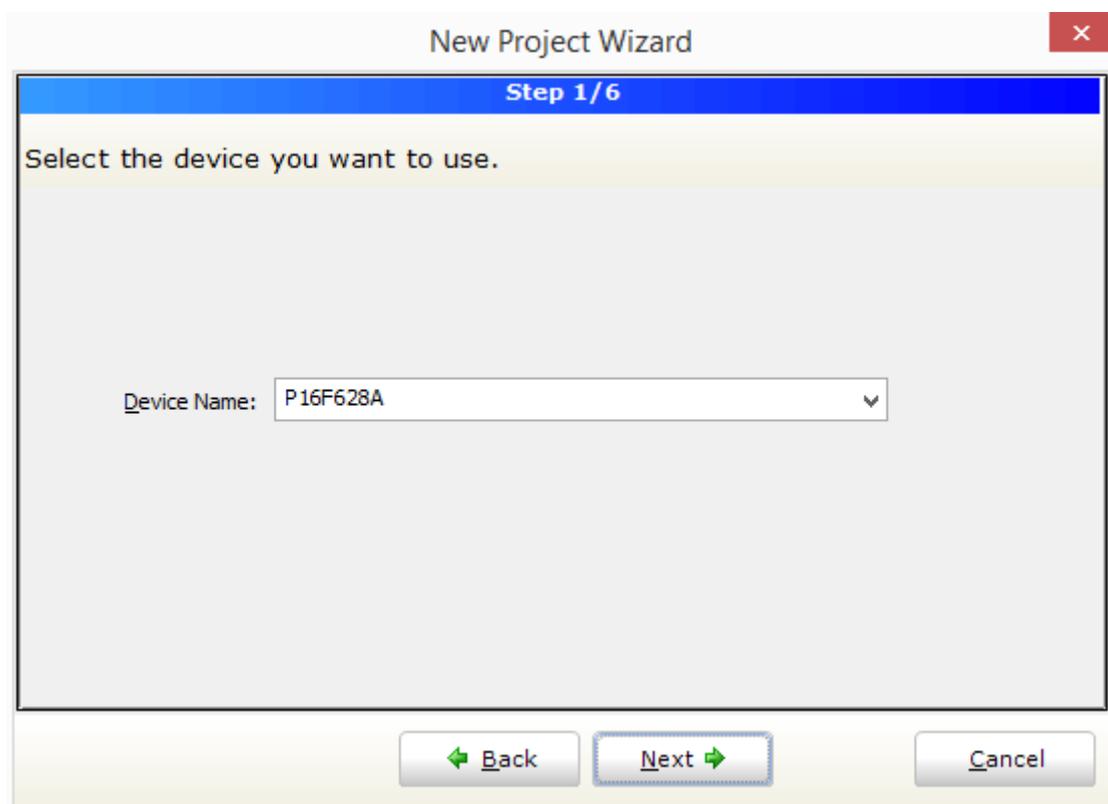
La meilleure façon de créer un projet c'est à l'aide de l'Assistant Nouveau projet (menu> Project> New Project) ou en cliquant sur l'icône Nouveau projet à partir de la barre d'outils du projet.



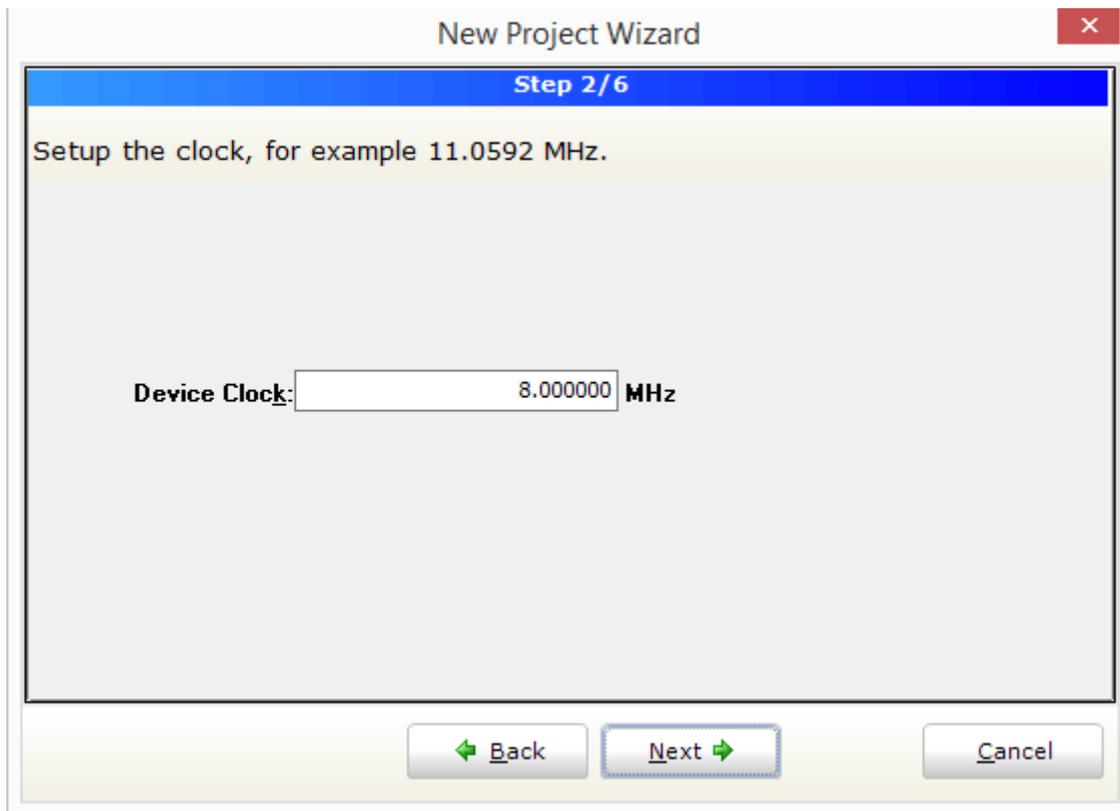
- Commencez à créer nouveau projet, en cliquant sur le bouton Next :



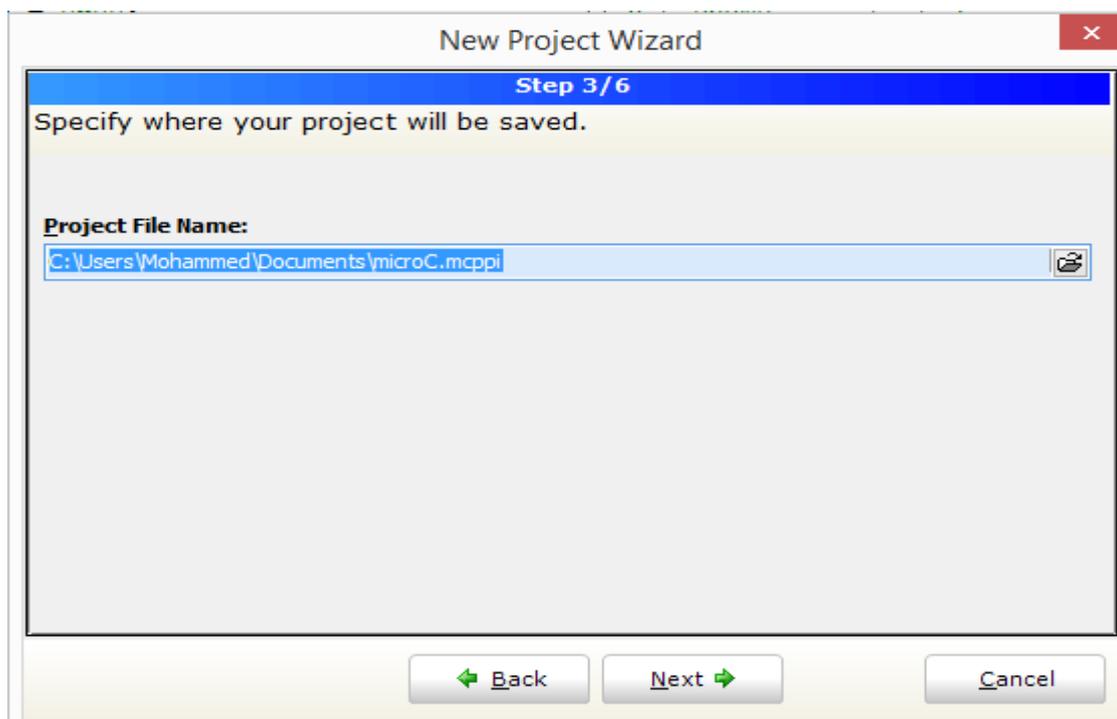
- Sélectionnez le périphérique dans « Device Name » dans la liste déroulante.



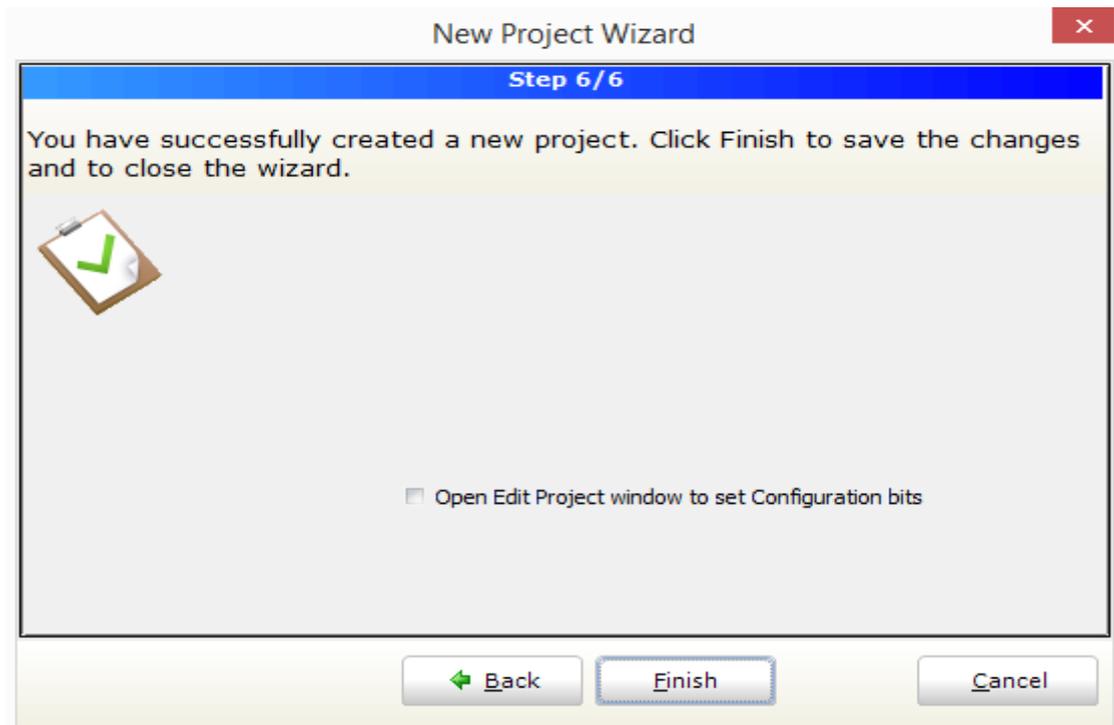
- Saisir la valeur de fréquence de l'oscillateur.



- Spécifiez l'emplacement où votre projet sera enregistré.

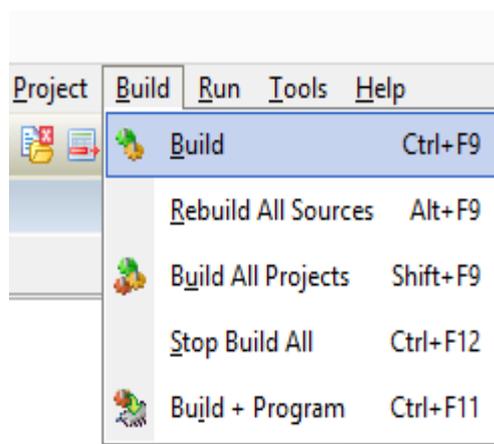


- Cliquez sur Finish pour créer votre nouveau projet.



Lorsque vous avez créé le projet et écrit le code source, il est temps de le compiler.

- Sélectionnez « Build » à partir du menu ou Ctrl F9.



La barre de progression s'affiche pour vous informer sur l'état de la compilation. S'il y a des erreurs, vous en serez informé dans la fenêtre d'erreur (Figure III-7).

Après que la compilation réussie, le compilateur mikroC PRO for PIC génère des fichiers de sortie dans le dossier du projet (dossier qui contient les fichiers projet. mcppi, .hex, .mcl, .lst, asm).

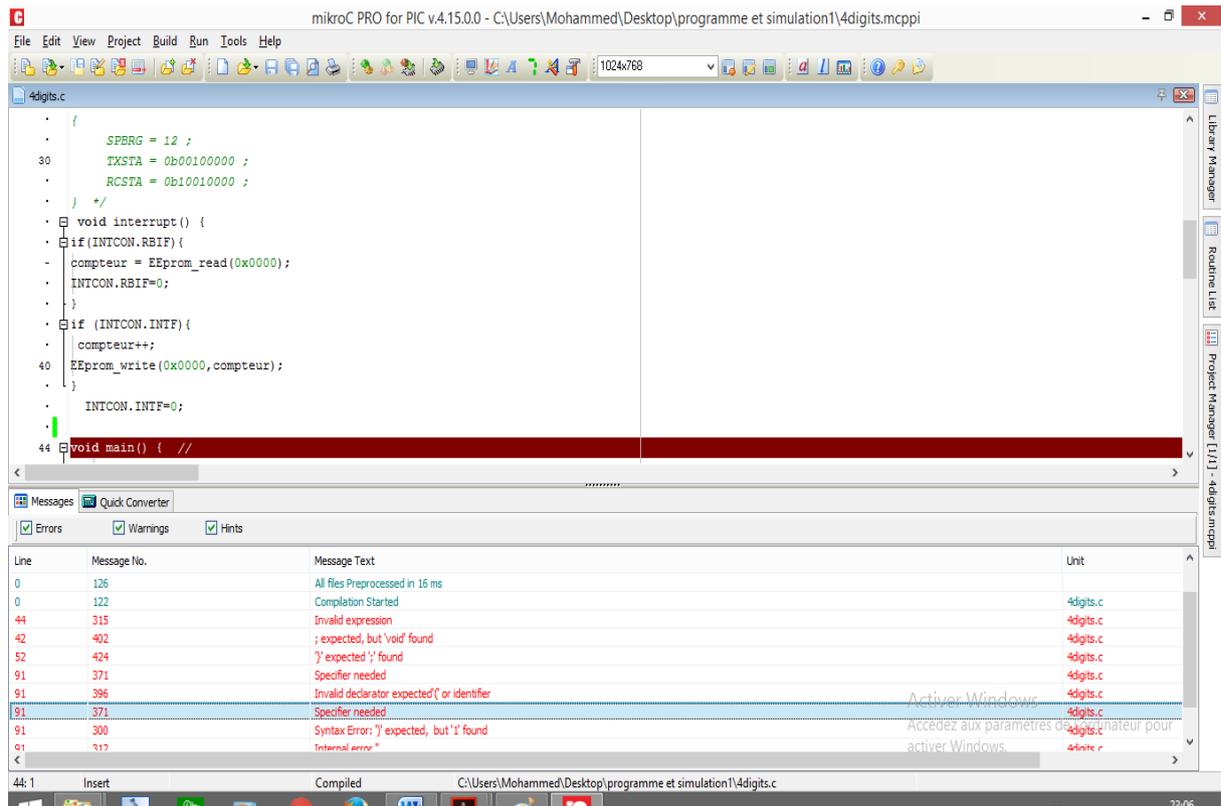


Figure III-11 : Avertissement des erreurs.

6.2 Isis Proteus Professional

Proteus est une suite logicielle permettant la conception assistée par ordinateur électronique éditée par la société Labcenter Electronics. Proteus est composé de deux logiciels principaux : ISIS, permettant entre autres la création de schémas et la simulation électrique, et ARES, dédié à la création de circuits imprimés.

Grâce à des modules additionnels, ISIS (Intelligent Schematic Input System) est également capable de simuler le comportement d'un microcontrôleur (PIC, Atmel, 8051, ARM, HC11...) et son interaction avec les composants qui l'entourent.

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^[13].

Il faut toujours prendre en considération que les résultats obtenus de la simulation sont un peu différents de celles du monde réel.

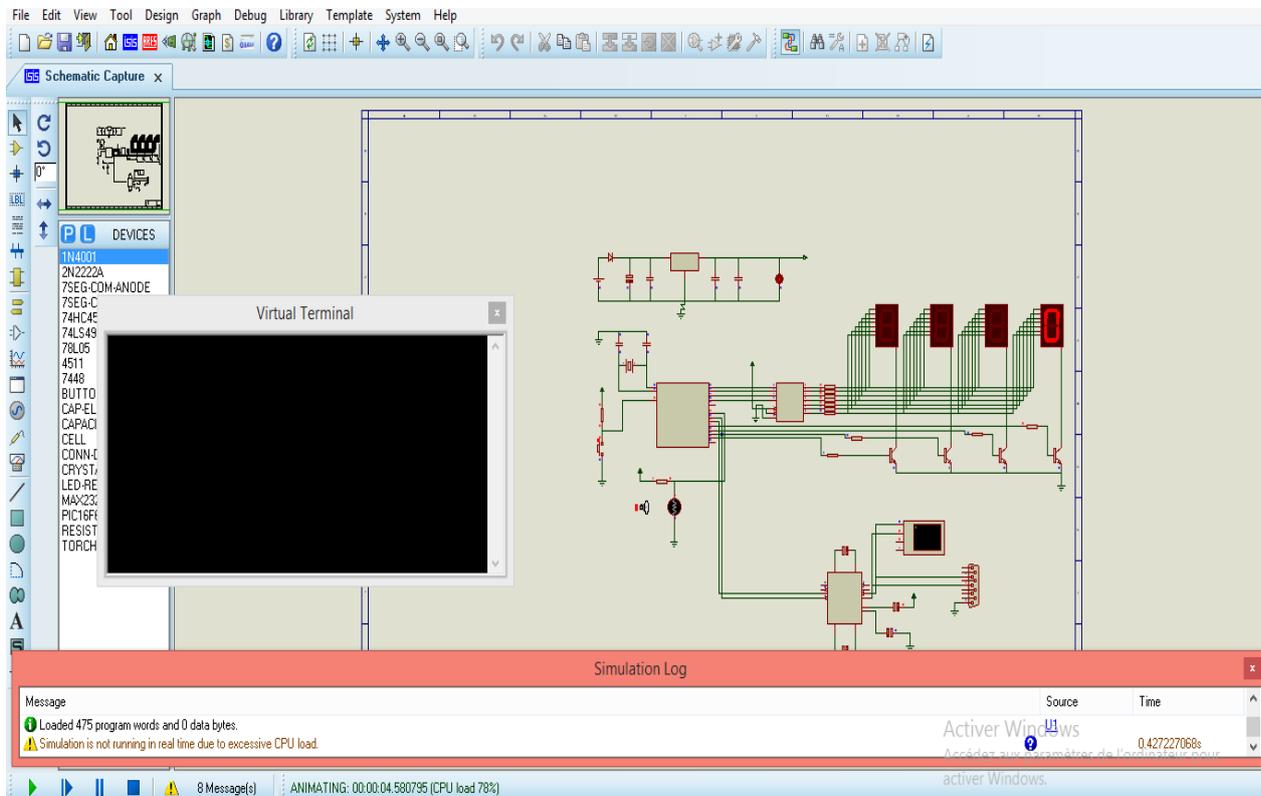
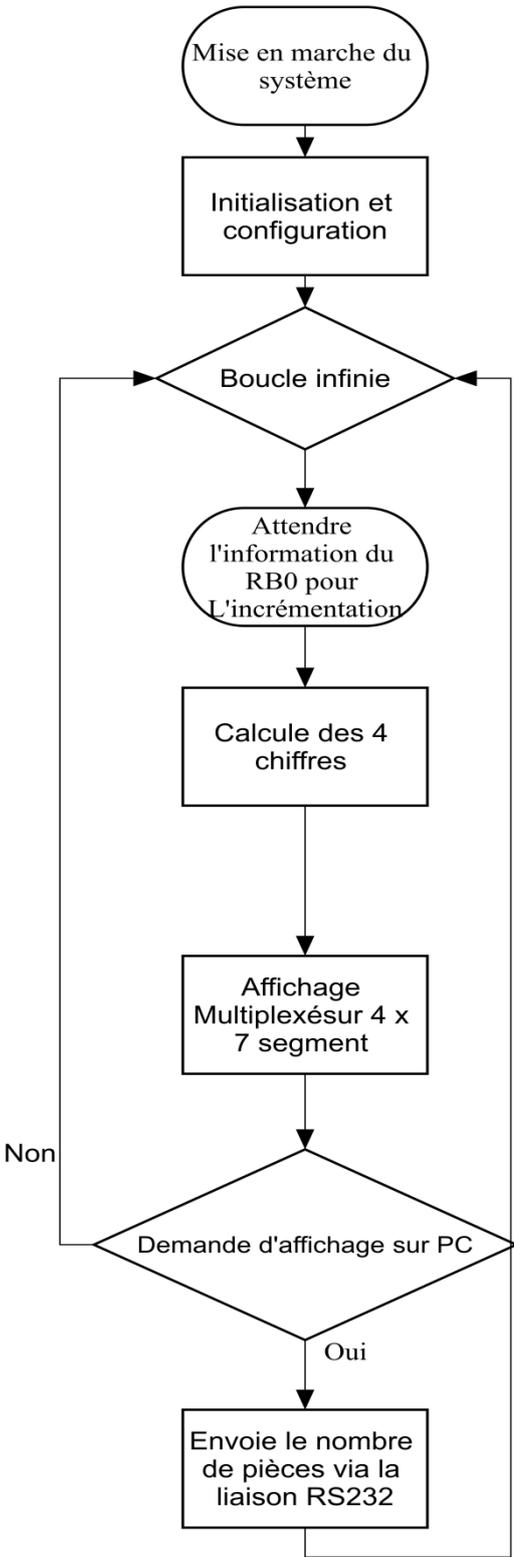


Figure III-12 : simulation avec Proteus.

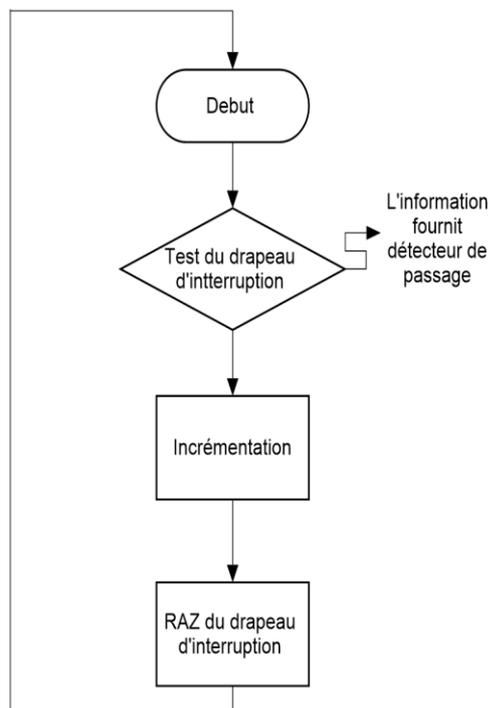
Tant que la fréquence d'affichage est élevée la simulation ne déroule pas dans le temps réel due à la charge excessive du CPU.

7. Programme embarquée dans le PIC

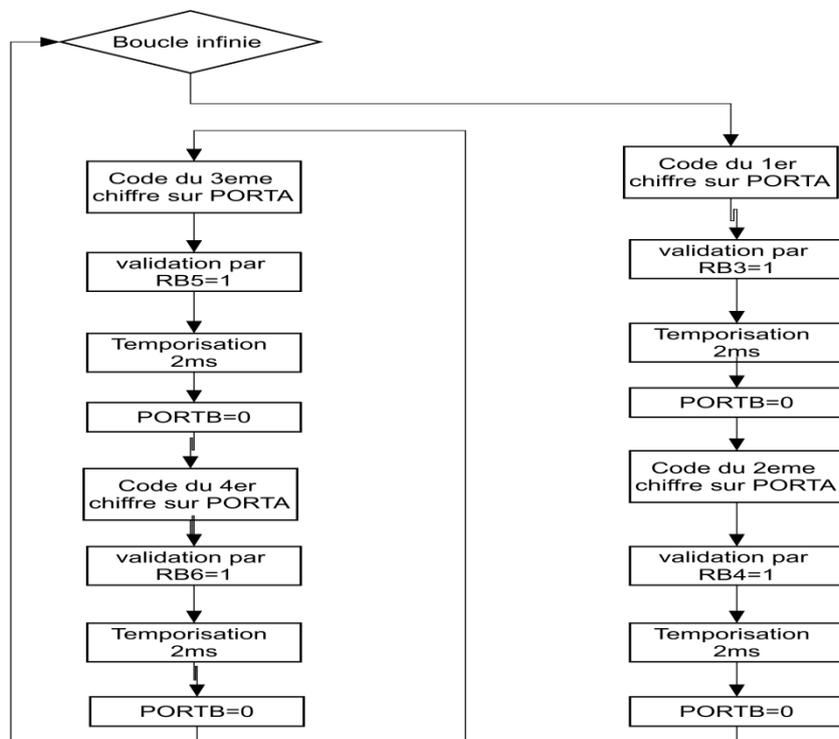
7.1 Organigramme Principale



7.1.1 Organigramme d'interruption



7.1.2 Organigramme d'affichage multiplexé



7.2 Description du fonctionnement de système

Ces trois organigrammes montrent le fonctionnement de notre programme.

Dans le premier, le principale, Après une mise en marche du système et configuration, dans une boucle infinie le système est en attente d'une interruption du détecteur pour incrémenter le registre de comptage et de l'envoyer en sortie vers l'afficheur et attendre une autre interruption pour l'envoi sur le port RB2/Tx vers la liaison RS232.

Dans le deuxième, c'est l'organigramme d'interruption, le système attend le drapeau d'interruption « INTF » pour incrémenter le compteur, qui égale 1 une fois qu'un changement d'état est détecté dans la broche RB0. Le drapeau doit être initialisé à zéro pour pouvoir utiliser l'interruption autre fois.

Dans le troisième, montre une commande multiplexée de quatre afficheurs cathode commune, les anodes des afficheurs sont commandées par le décodeur BCD ; les cathodes des différents afficheurs sont reliées à des transistors NPN et pilotée par le PORTB (RB3 :RB6). Pour faire apparaître un chiffre sur l'afficheur 1, le PIC place le code BCD du chiffre à afficher sur les lignes RA0 à RA3 et porte à 1 la ligne RB3 les autres restants à 0. Seuls les segments de l'afficheur 1 s'allumeront puisque leurs anodes sont à 5Volts et leurs cathode est tirée vers la masse par le transistor. Après 2 millisecondes l'afficheur 2 s'allume, donc on fait le tour de tous les afficheurs. Si chaque 'N' afficheur est allumé 'n' millisecondes, le temps d'un balayage complet vaut « n x N » et la fréquence vaut « 1/ (n x N) » => f = 125 Hertz.

7.3 Initialisation et configuration

Pour pouvoir utiliser les différents ports du PIC, MikroC exige qu'on lui indique les entrées et les sorties dans chaque port :

```
TRISB=0b00000111; // configuration du RB0,RB1,RB2 entré et le reste en sortie
```

```
TRISA=0; // configuration du PORTA en sortie
```

```
PORTB=0;PORTA=0; // initialisation à zéro.
```

```
CMCON=0b00000111; // désactivation du comparateur pour utiliser RA3 et RA4  
en mode numérique.
```

7.4 Les fonctions et les variables

```
unsigned char i, *a="0000";
unsigned int compteur,num, millier, centaine, dizaine, unite; // déclaration des variables
```

Notons que pour l'affichage d'un nombre composé de 4 chiffres il faut convertir et envoyer chaque chiffre tout seul :

```
void calchiffre(){

    millier = (compteur/1000) %10 ; // Calcul de la valeur des milliers. On divise le
    compteur par 1000 on prend le modulo

    centaine = (compteur/100) %10; // calcul de la valeur des centaines. On divise le
    compteur par 100 on prend le modulo

    dizaine = (compteur/10) %10; // calcul de la valeur des dizaines. On divise le
    compteur par 10 on prend le modulo

    unite = compteur %10; } // calcul de la valeur des unités. On prend le modulo
```

7.4.1 Fonction d'initialisation du module USART :

La liaison RS232 est caractérisée par sa vitesse de transmission en bauds par seconde : 300, 1200, 2400, 4800, 9600, 19 200, 38 400 etc ...

Il est bien clair que la vitesse de réception est identique à la vitesse de transmission. Le choix se fait avec : le registre 8 bits SPBRG et bit 2 du registre TXSTA, BRGH.

La valeur du registre SPBRG calculé par la formule suivante :

Si BRGH = 1 : $SPBRG = (Fosc / (16 \times \text{Vitesse de transmission})) - 1$

BRGH = 0 : $SPBRG = (Fosc / (64 \times \text{Vitesse de transmission})) - 1$

La vitesse de transmission souhaitée est égale à 9600 :

Si BRGH = 1 : $SPBRG = (8\ 000\ 0000 / (16 \times 9600)) - 1 = 51$

Si BRGH = 0 : SPBRG = (8 000 0000 / (64 x 9600)) - 1 = 12

=> Vitesse de transmission = 8 000 000 / (16(51 + 1)) = 9615 bauds par seconde

=> 0,16 % d'erreur ce qui est satisfaisant pour les 2 valeurs.

```
void usart_init() { // fonction d'initialisation du module USART

SPBRG = 12 ; // bauds 9600

TXSTA = 0b00100000 ; //lance de transmission sur 8 bit

RCSTA = 0b10010000 ; } // mise en service de l'USART et autorisation de la réception série
```

Fonction pour le saut de ligne dans l'application HyperTerminal :

```
void newline(){

UART1_write(13); // retour à la ligne

UART1_write(10); } // charge de ligne
```

7.5 Gestion d'affichage : 4 x 7 segments et dans l'application HyperTerminal

```
while(1) { // boucle infinie
void calchiffre(); // l'appelle du fonction cal chiffre
PORTA = millier; // l'envoi du chiffre millier dans le port A
PORTB.B6=1 ; // On active le 4ième digit en mettant RB6 à 1
delay_ms (2); // On marque une pause pour afficher le chiffre
PORTB=0;
PORTA = centaine;
PORTB.B5=1 ;
delay_ms (2);
PORTB=0;
PORTA = dizaine;
PORTB.B4=1;
delay_ms (2);
PORTB=0;
```

```

PORTA = unite;
PORTB.B3=1;
delay_ms (2);
PORTB=0;
if(UART1_Data_Ready())           // test de reception
i=UART1_Read();                  // met le code ascii reçue dans i
if(i='n'){                       // si i égale le code ascii du « n »
i=0;
a[0] = millier + 48;             // conversion des valeurs en code ascii
a[1] = centaine+48;
a[2] = dizaine+48;
a[3] = unite+48;
UART1_Write_Text("nombre de pieces est:");
uart1_write_text(a); newline();} // transmission de la valeur du compteur vers pc

```

7.6 Interruption

Pour utiliser les interruptions, il faut tout d'abord les activer. Il faut pour cela fixer les valeurs de registre INTCON:

```

INTCON = 0b10010000;           // activation d'interruption RB0
void interrupt() {             // interruption du RB0 pour incrémentation du compteur
if (INTCON.INTF){
compteur++;}
INTCON.INTF=0;} // RAZ le drapeau d'interruption pour l'utiliser chaque fois.

```

8. Conclusion :

Dans ce chapitre nous avons mis en œuvre les différentes parties du système et l'implantation du programme. Au-delà l'acquisition des informations nécessaires pour le câblage des composants électronique qui constituent notre système de comptage.

Dans le chapitre suivant nous montrerons comment le programme sera chargé dans la mémoire du microcontrôleur. Nous procéderons aussi à la simulation et le test de notre programme: clignotant d'une LED, l'affichage multiplexé 4 x 7 segment, la carte de communication RS232 jusqu'au montage finale.

Chapitre

IV

Réalisation

Et Expériences

1. Introduction

Après l'implémentation et la compilation du programme, il doit être chargé dans la mémoire du PIC 16F628. Pour programmer un microcontrôleur PIC, il est nécessaire de posséder un programmeur. De nombreux kits sont disponibles sur le marché, Cependant, certains ne permettent de programmer qu'une catégorie de PIC, il est donc important de bien se renseigner avant l'achat.

2. Programmeur Universel USB - Mini Pro TL866CS

TL866CS est un programmeur universel conçue par Autoelectric, qui est capable de supporter ATMEL89S51, 52, toute série ATMEGA- AVR et toute la série MICROCHIP PIC10F, 12F, 16F, 18F, SYNCMOS SM59Dxx / SM59Rxx gamme complète. Mini Pro fait partie des plus rapides programmeurs universels du marché avec haute performance et une interface USB de grande capacité, à la vitesse de communication de 12Mbps [16].



Figure IV-1 : Programmeur Universel USB - Mini Pro TL866CS.

Ce programmeur est joint par une interface de développement dans un CDROM dite « MiniPro V6.00 » qui fonctionne sur toutes les plateformes, y compris 32/64 bit Windows 8.

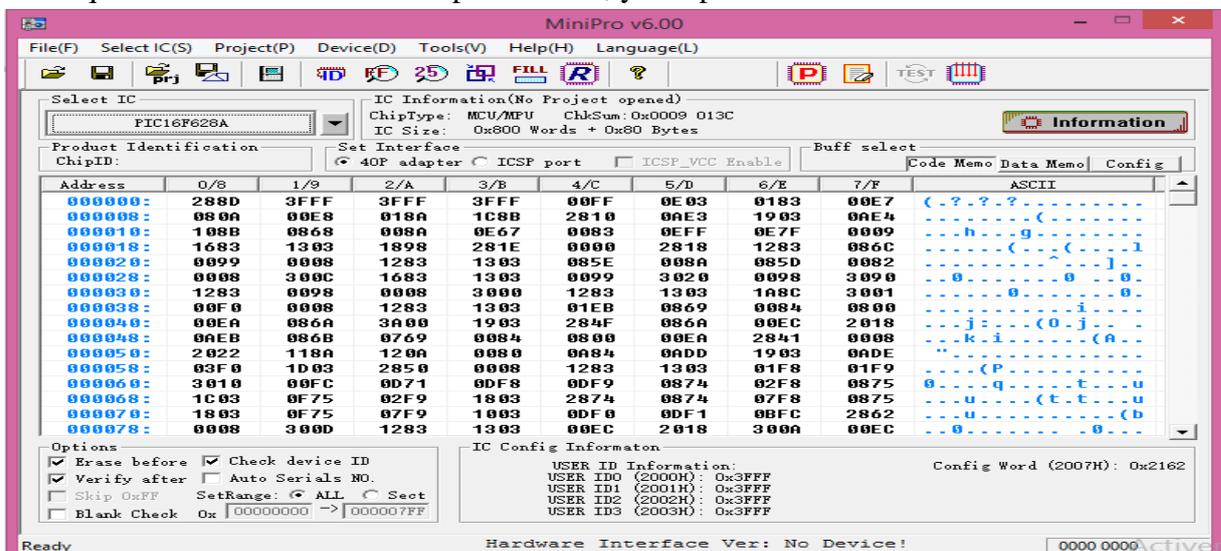
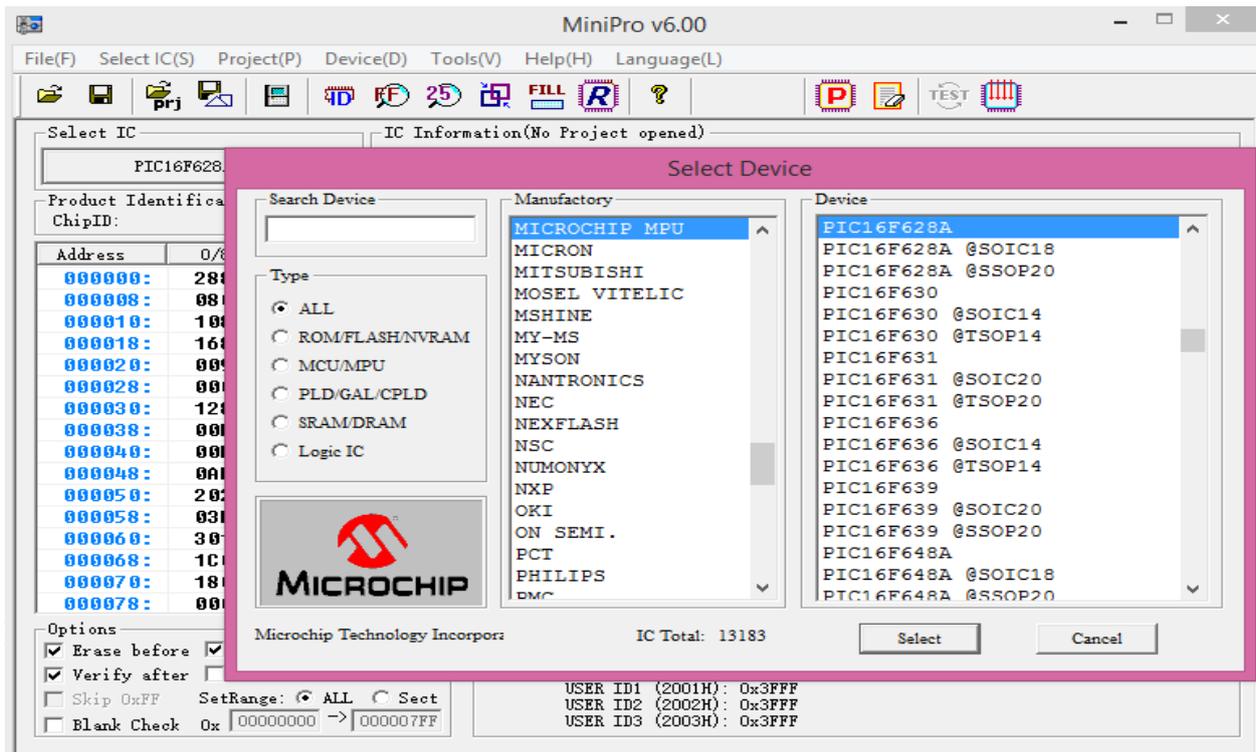
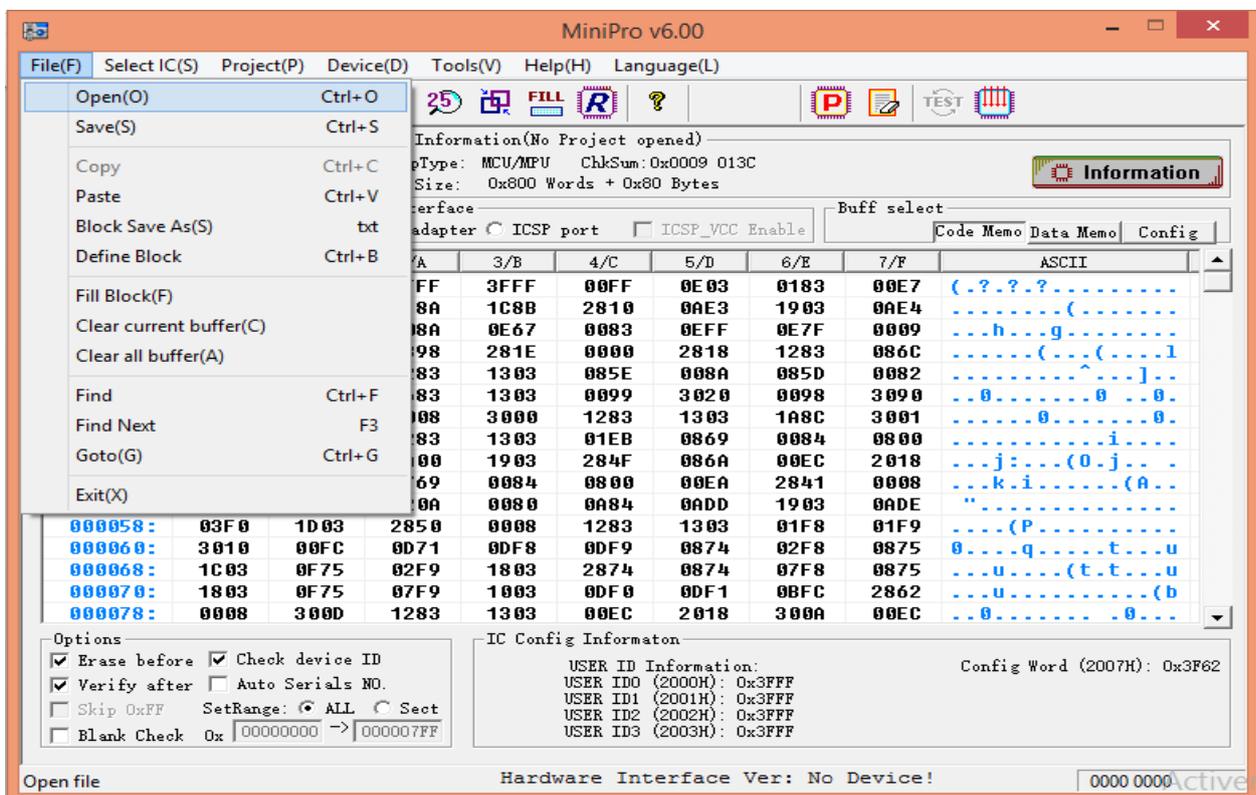


Figure IV-2 : L'IDE MiniPro V6.00

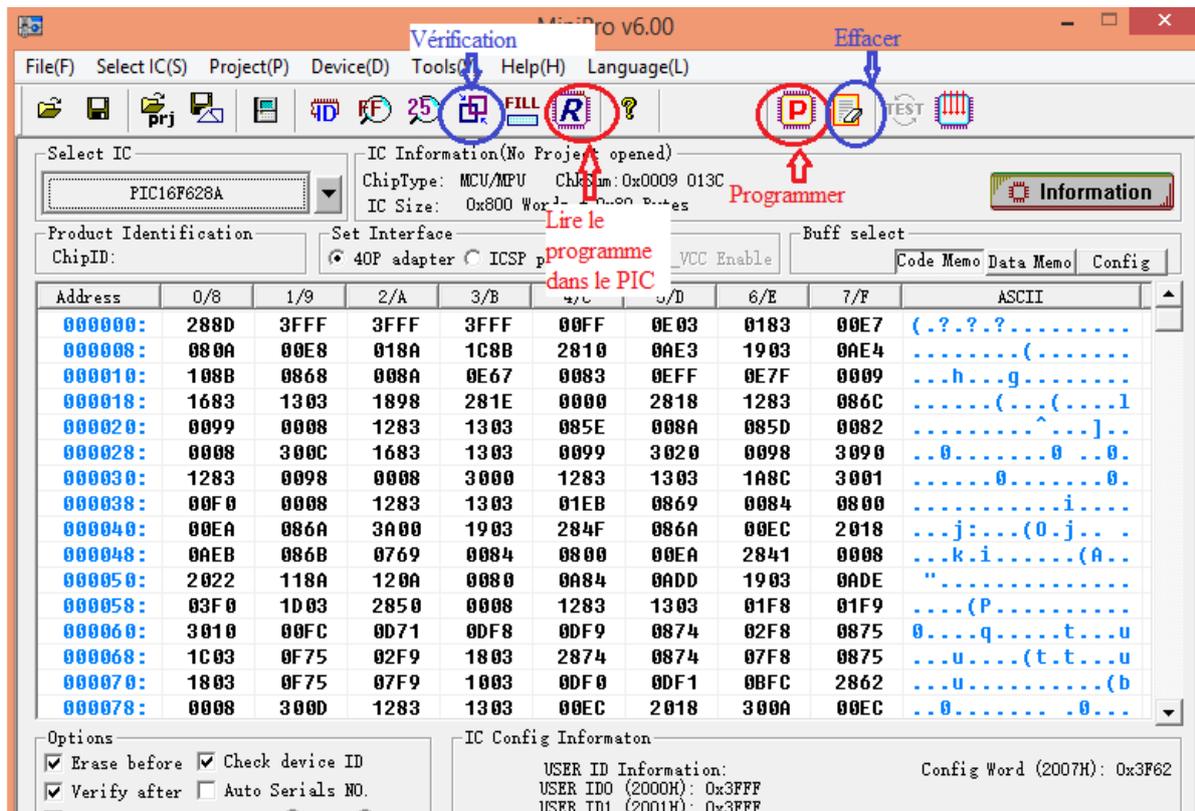
Évidemment, il faut connecter le matériel à l'USB et sélectionnez le PIC 16f628.



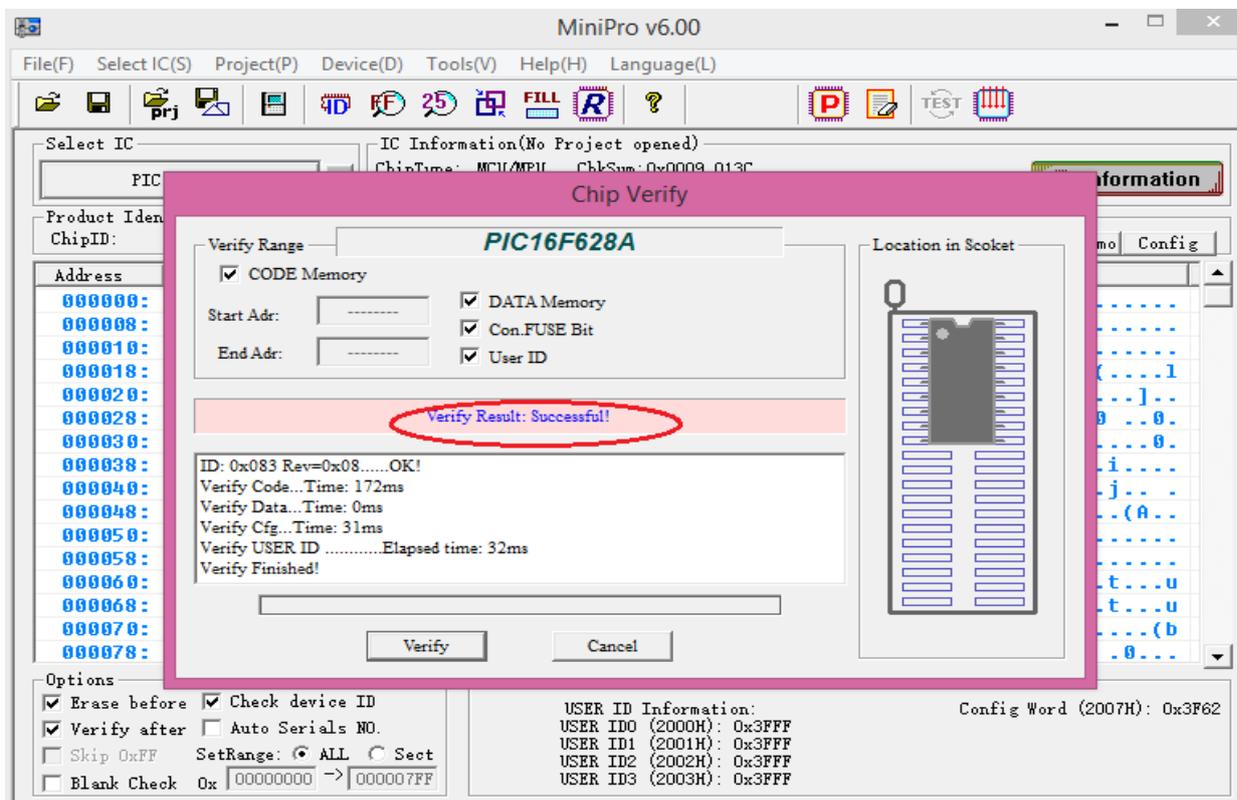
Pour charger le fichier HEX sur le tampon, il faut aller sur « File » => à « Open » et accéder à l'emplacement où le fichier HEX, qui doit être fusionné dans le PIC, est enregistré.



Les options de chargement du programme sont les suivant :



Le message « Verify Result: Successful! » apparait si la fusion est terminée.



3. Applications par le PIC 16f628

3.1 Commande d'un afficheur 7 segments

Cette application consiste à visualiser les nombres décimal de 0 à 9 sur l'afficheur 7 segments cathode commune. Un bouton poussoir permet d'incrémenter le comptage sur l'afficheur.

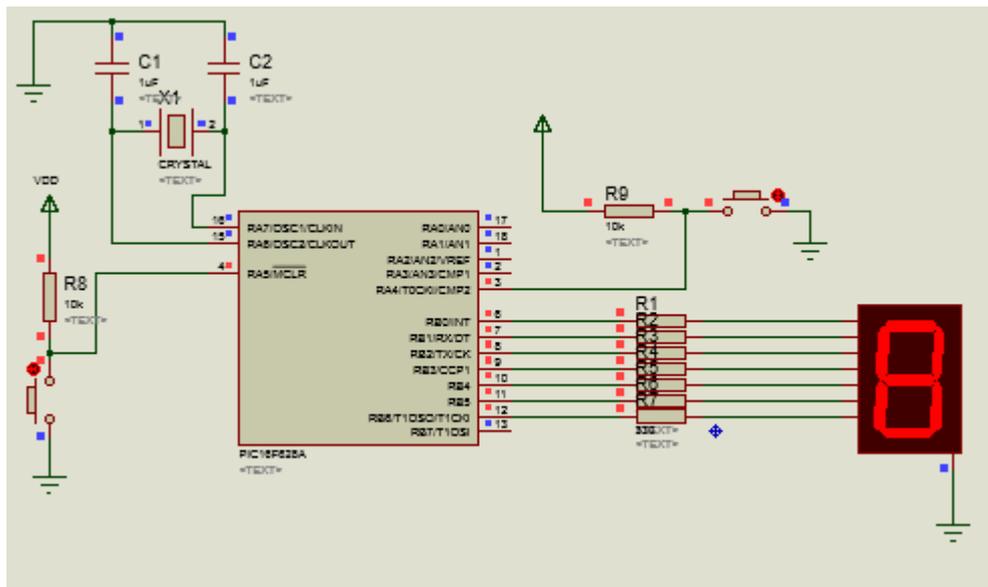


Figure IV- 3 : Simulation de la commande d’afficheur par ISIS.

3.1.1 Le code C et les commentaires

```
unsigned int num;
```

```
// Utilisation de la commande "switch" qui compare la valeur contenu dans "num" au
valeur "case" ci-dessous
```

```
switch (num) {
```

```
// Si une valeur est trouvée, "Switch" renvoie la valeur correspondante et sort, une seule
valeur est prise en compte
```

```
case 0 : return 0x3F; // "0" l'allumage de six segments 3F = 00111111
```

```
case 1 : return 0x06; // |
```

```
case 2 : return 0x5B; // |
```

```
case 3 : return 0x4F;    //
case 4 : return 0x66;    //
case 5 : return 0x6D;    //
case 6 : return 0x7D;    //
case 7 : return 0x07;    //
case 8 : return 0x7F;    //
case 9 : return 0x6F;    //  }}  |
```

/ Cette fonction retrouve la configuration de bits à envoyer au port B pour afficher un numéro sur la LED 7 segments. Le numéro est passé dans la liste des arguments de la fonction */*

```
void main() {
```

```
TRISB=0b00000000; // configuration du PORTB en sortie
```

```
TRISA=0b00010000; // configuration RA4 en entrée
```

```
PORTB=0; PORTA=0;
```

```
CMCON=7; //désactivation du comparateur et configurer RA3 et RA4 en E/s numérique
```

```
OPTION_REG=0b00111000; // Taux de prédiviseur est de 1:1 // TMR0 reçoit des impulsions par la broche RA4.
```

```
TMR0=0; // Initialisation du compteur TMR0
```

```
do{ // BOUCLE INFINIE
```

```
PORTB = conversion (TMR0); // met la valeur du TMR0 dans PORTB
```

```
if (TMR0==10) {
```

```
TMR0=0; } // initialisation du TMR0 à zéro avant qu'il atteint 10
```

```
} while(1); }
```

3.1.2 Câblage de la plaque d'expérimentation

- Un quartz de 8 MHz pour cadencer le PIC, avec deux condensateurs de 22 pf.
- Une résistance de 330 Ohms doit être insérée avec les 7 segments pour limiter le courant dans l'afficheur.
- Une résistance de rappel 10 KOhms impose un 5 Volt sur l'entrée RA4 reliée à l'interrupteur et une autre permet un RAZ à la mise sous tension sur la broche MCLR du PIC.
- Afficheur 7 segments relié avec PORTB.

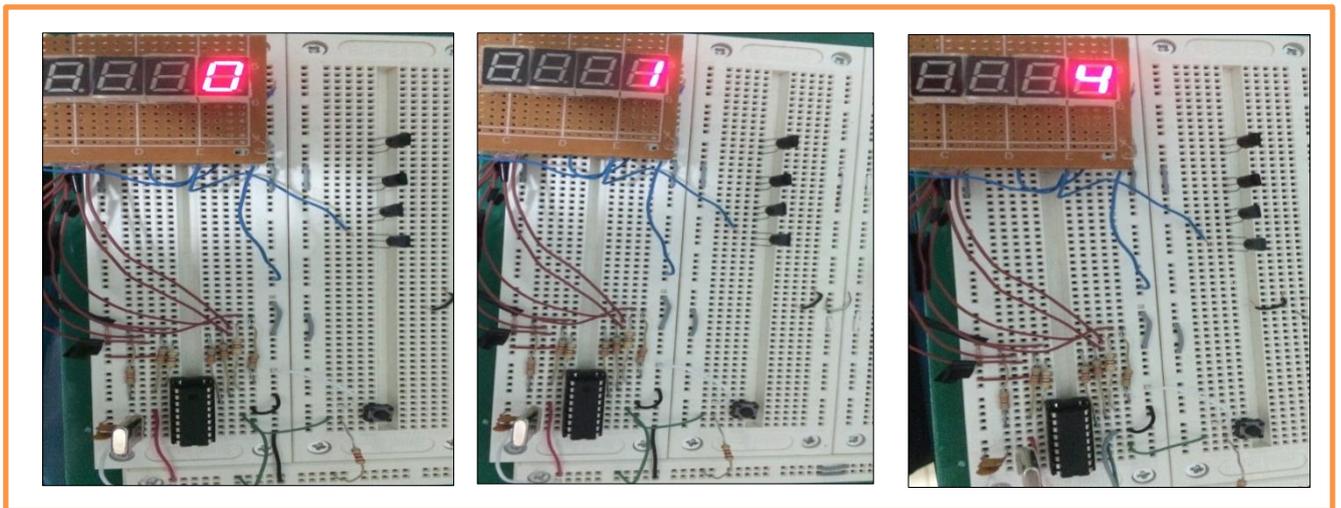


Figure IV- 4 : Le montage sur la plaque d'expérimentation d'afficheur 7 segments.

3.2 Test du module USART

Afin de faire une liaison série RS232 on a réalisé une petit plaque avec le circuit intégré MAX232 et un connecteur femelle type DB-9 pin.

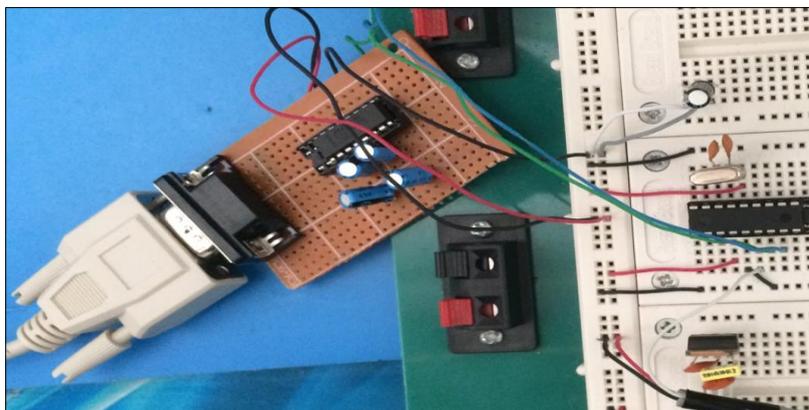


Figure IV- 5 : La liaison série RS232 entre PIC et PC.

3.2.1 Commentaire et Code C

Comme d'habitude, le programme est développé avec le MikroC Pro pour compilateur PIC. Le compilateur MikroC fournit UART bibliothèque qui prend en charge la communication série asynchrone en mode duplex intégral (ce qui signifie transmettre et recevoir simultanément). Cela rend le lot de programmation plus facile. Par exemple, pour envoyer un texte par le module USART de PIC16F628, il faut juste écrire `UART1_Write_Text ("Texte")`. Le code donné ici établit une liaison série asynchrone double sens entre le microcontrôleur PIC16F628A et le PC. Le microcontrôleur envoie le message « Appuyer sur une touche » qui est affiché sur la fenêtre HyperTerminal. Lorsqu'un caractère est rentré à partir du clavier, il sera envoyé au microcontrôleur via le port COM. Le microcontrôleur PIC le lira et l'enverra à l'ordinateur, qui sera à nouveau affiché sur la fenêtre HyperTerminal.

Les paramètres d'application HyperTerminal devraient être configurés en même que les paramètres du module USART :

Débit : 9600, Bits de données: 8, Parité: Aucune, Bits d'arrêt: 1, contrôle de flux: Aucun.

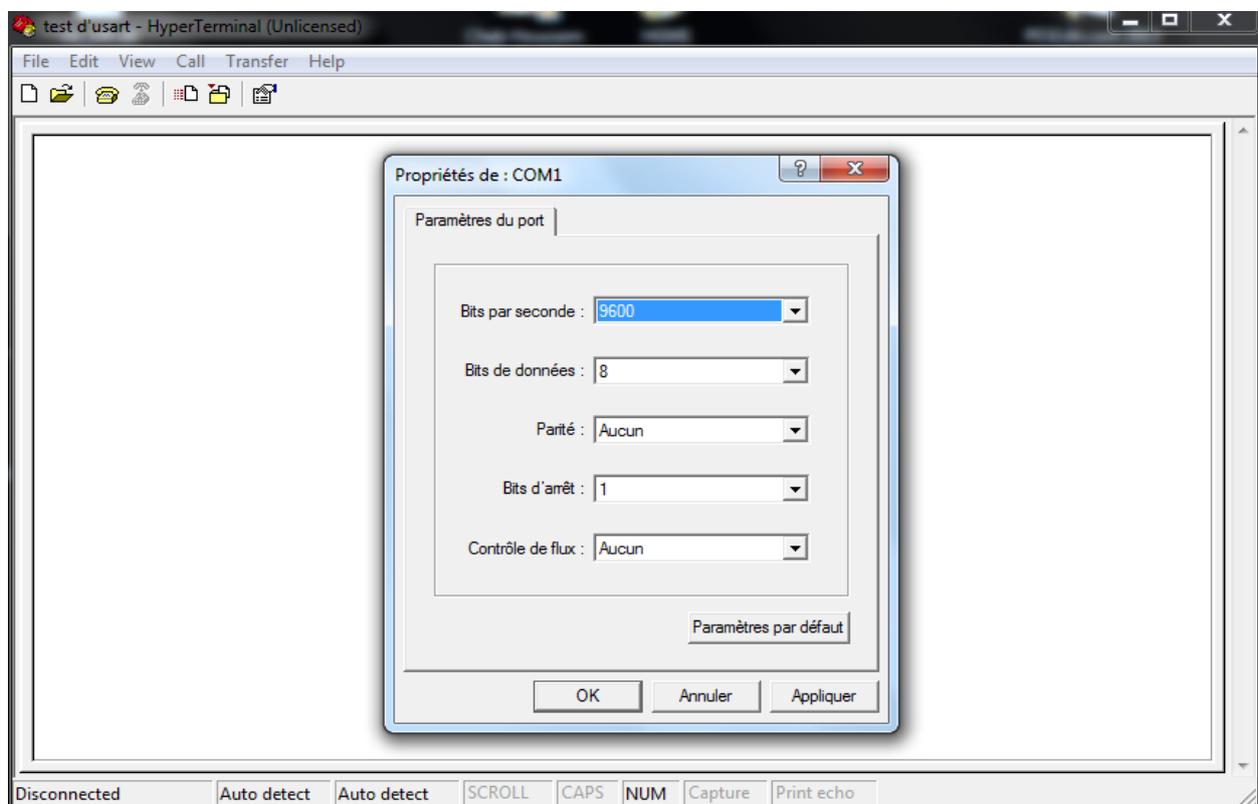


Figure IV- 6 : Configuration des paramètres d'HyperTerminal.

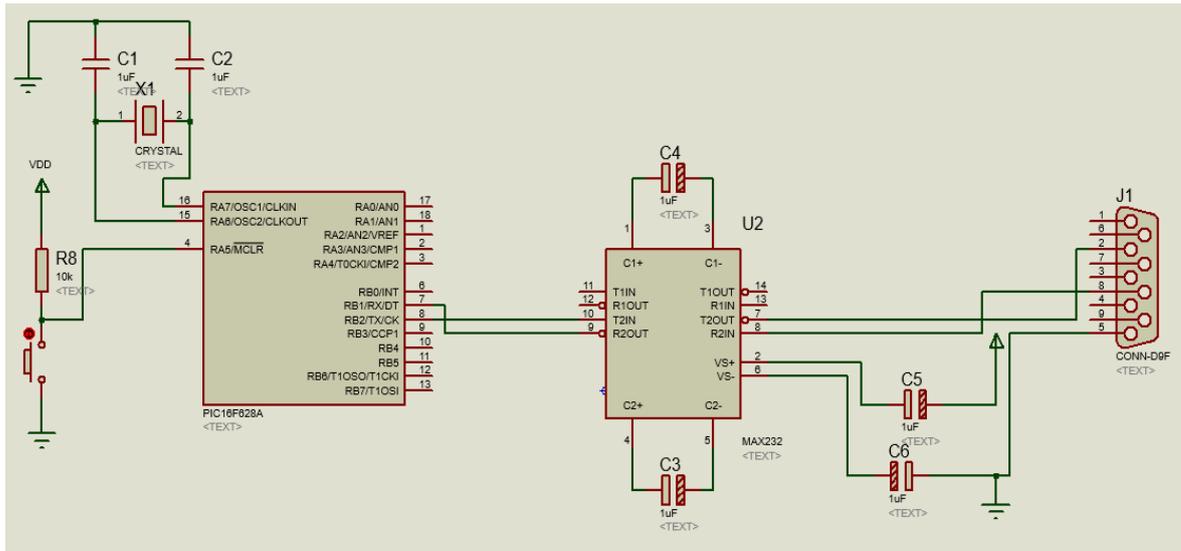


Figure IV- 7 : Schémas de la liaison RS232 et Simulation

Le code C :

```

void usart_init() { // fonction d'initialisation de l'USART

SPBRG = 12 ; // bauds 9600

TXSTA = 0b00100000 ; //lance de transmission sur 8 bit

RCSTA = 0b10010000 ; // mise en service de l'USART et autorisation de la réception série

}

void newline(){

UART1_Write(13); // Retour à la ligne

UART1_Write(10); // charge de la ligne

}

void main() {

unsigned char Temp;

TRISB = 0b00000110;

usart_init();
    
```

```
Delay_ms(100);

UART1_Write_Text("Testing USART ");

newline();

do {

UART1_Write_Text("Appuyer sur une touche ");

while(!UART1_Data_Ready());

Temp = UART1_Read();

newline();

UART1_Write_Text(" tu entre:");

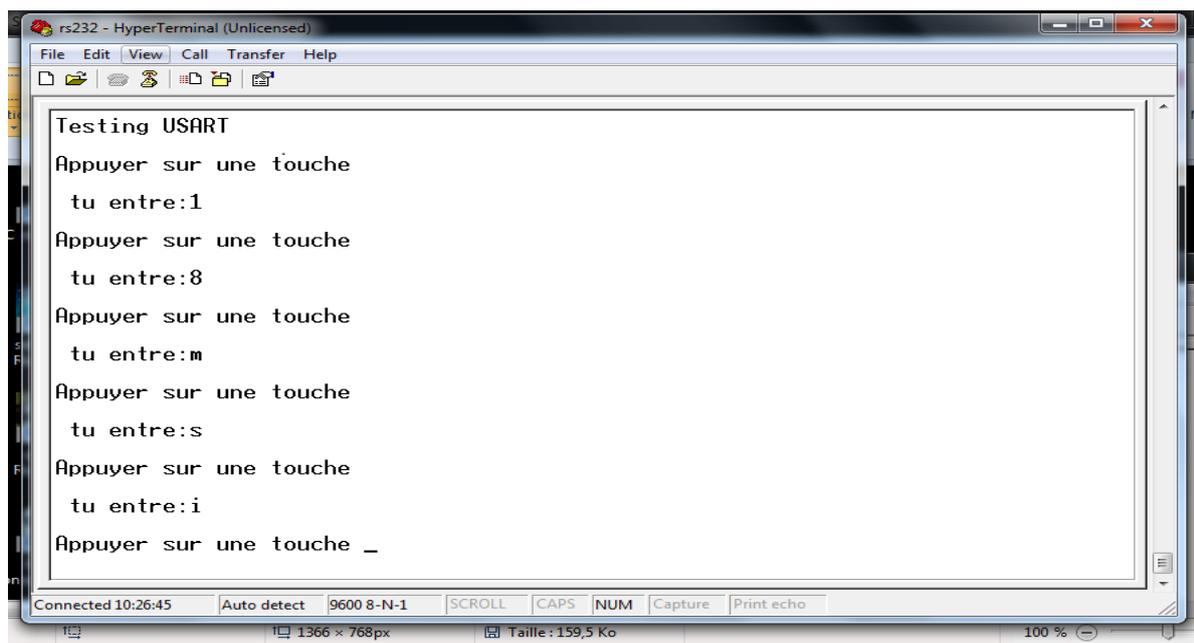
UART1_Write(Temp);

newline();

} while(1);

}
```

Voici un aperçu de la sortie de la fenêtre HyperTerminal sur Windows XP.



4. Réalisation de notre Projet

4.1 Liste des composantes

- PIC 16f628 ;
- Une pile 9 volt et un régulateur 5 Volts LM7805 ;
- 2-Condensateurs 270uf ; 2- 1uf chimique ; 1- 100uf céramique ; pour filtrage de la tension ;
- 4- Transistor 2N2222 NPN et 4 résistances 10 KOhms pour le pilotage des afficheurs ;
- MAX232, 4- condensateur 1uf ; connecteur DB-9 femelle et un câble pour la liaison RS232 ;
- Décodeur 74LS48 BCD to 7 segments.
- Afficheurs 4 x 7 segments et 7 résistances 330 ohms pour limitations de courants dans les segments des afficheurs ;
- Diode laser, LDR et une résistance 10KOhms afin de réaliser le capteur.
- Botton poussoir et une résistance 10KOhms pour le RESET du PIC.
- Quartz 8Mhertz pour cadencer le PIC et 2 condensateurs 22 pF.

4.2 câblage de la plaque d'expérimentation

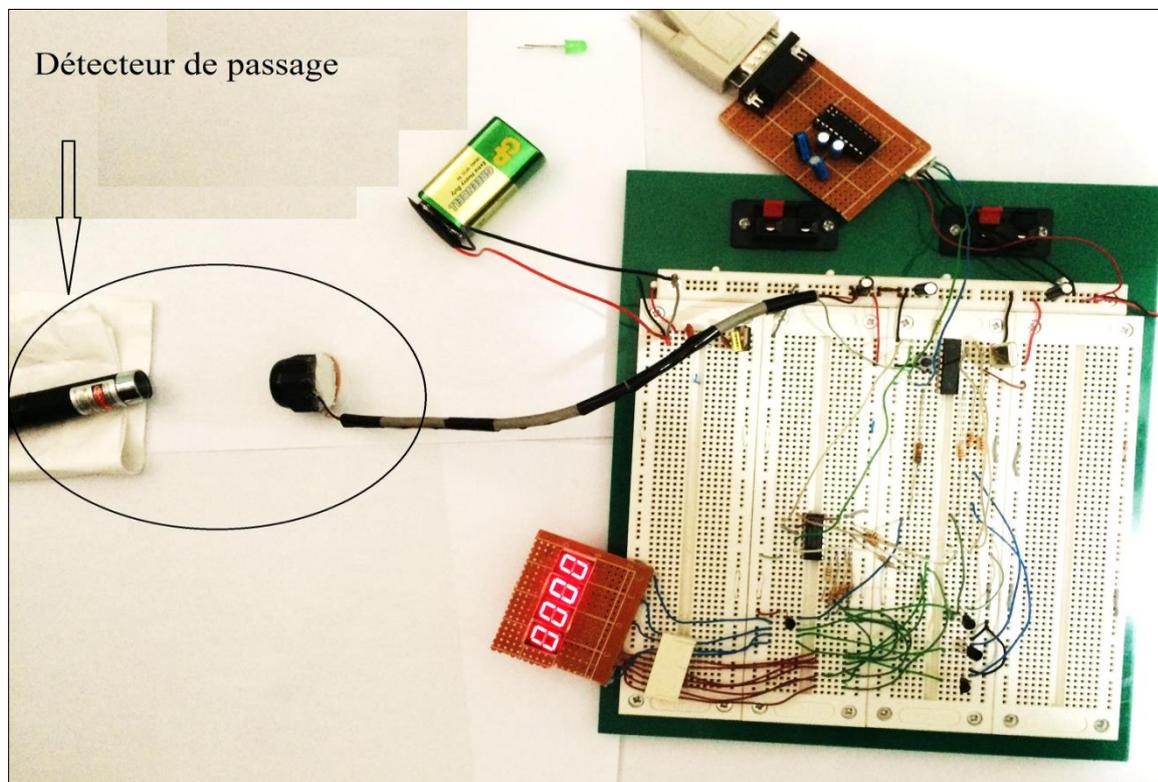
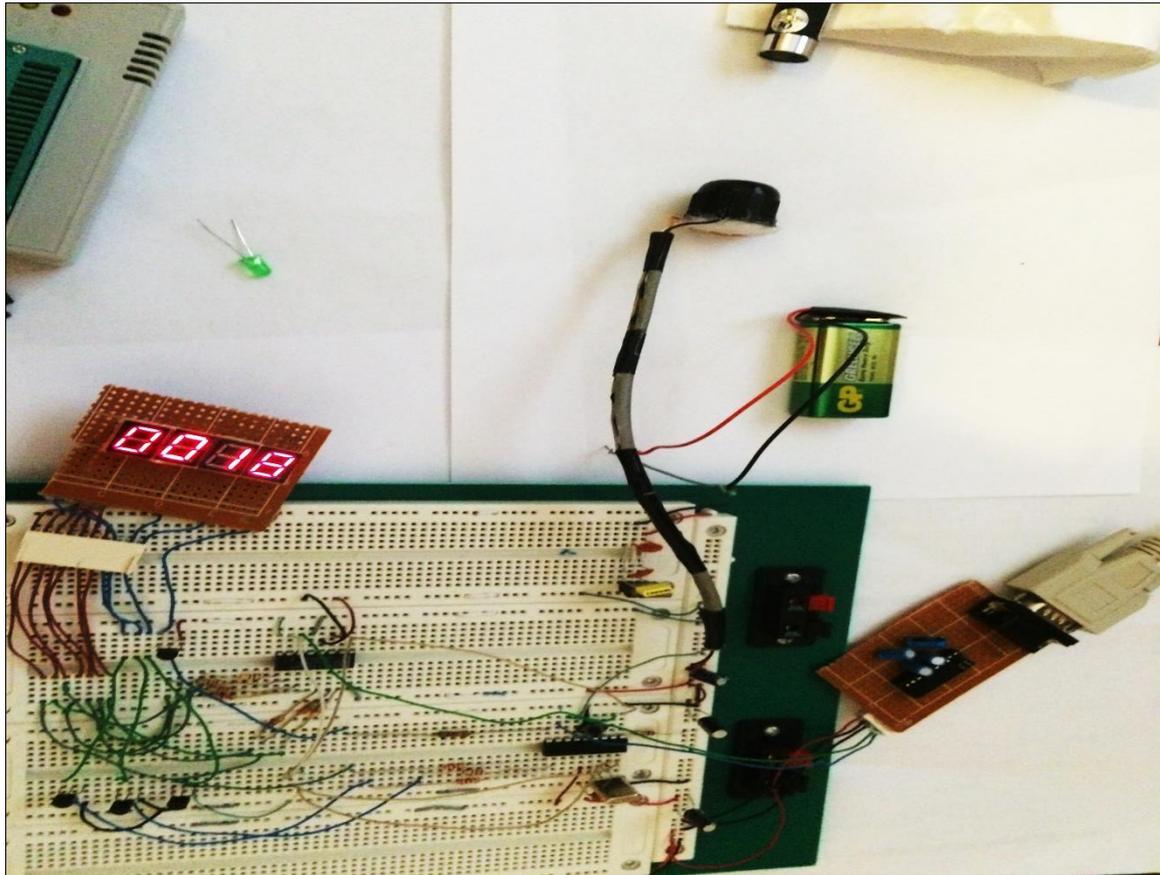
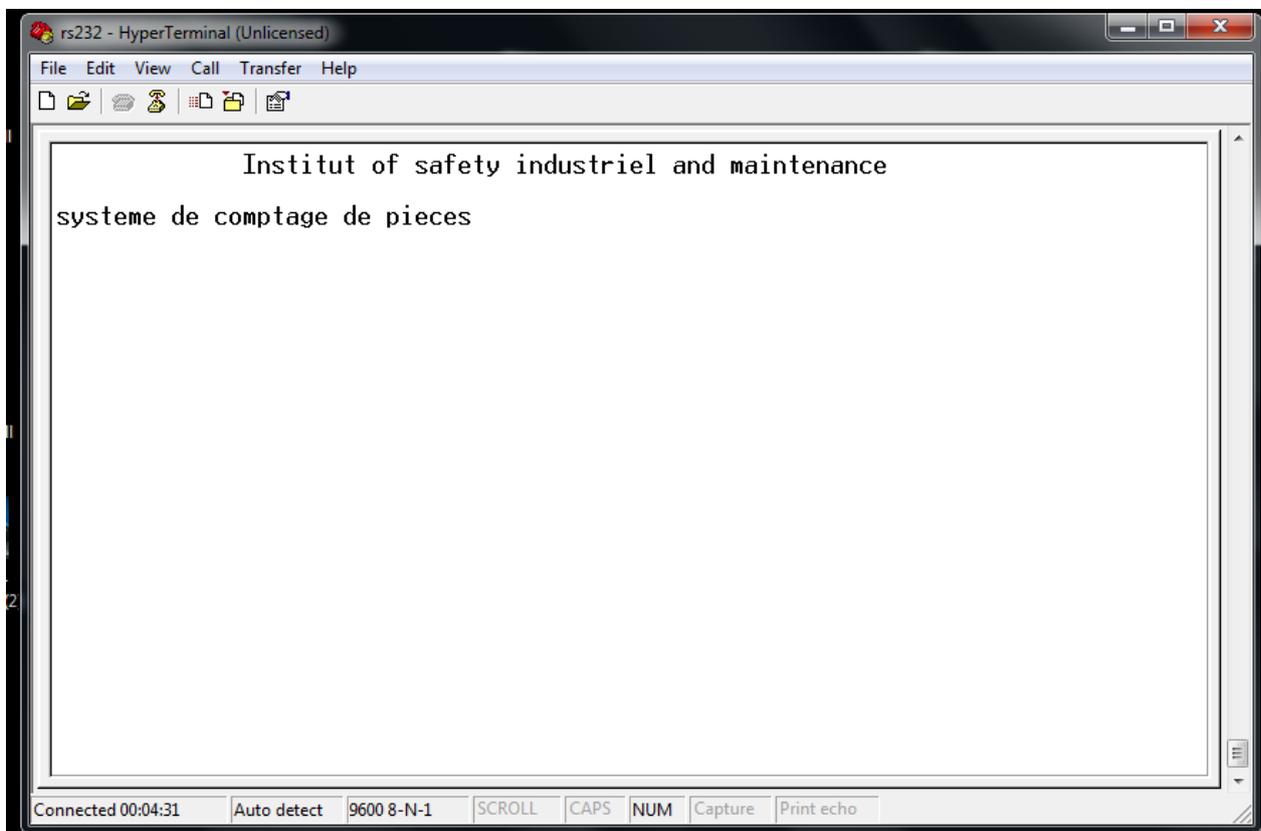
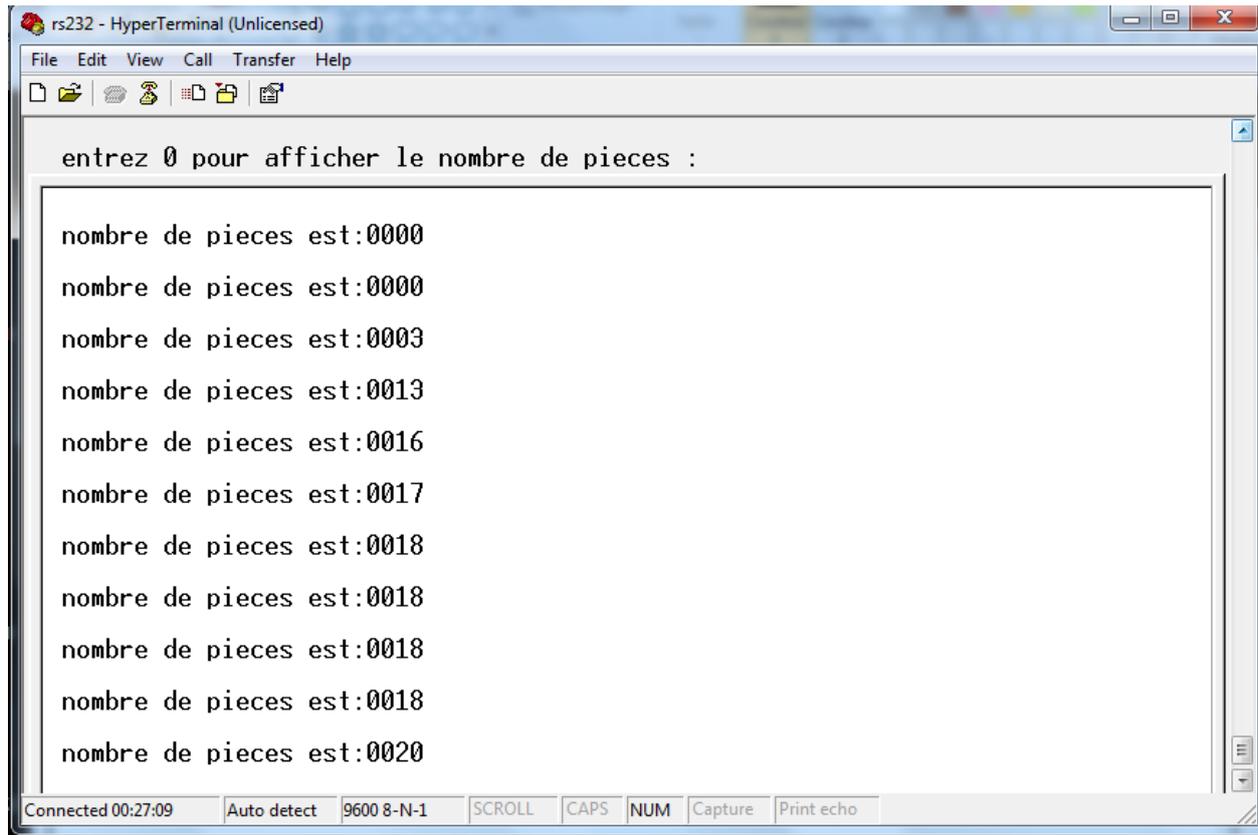


Figure IV-8 : Montage de système dans la plaque d'expérimentation



Affichage sur PC : application HyperTerminal





```
rs232 - HyperTerminal (Unlicensed)
File Edit View Call Transfer Help
entrez 0 pour afficher le nombre de pieces :
nombre de pieces est:0000
nombre de pieces est:0000
nombre de pieces est:0003
nombre de pieces est:0013
nombre de pieces est:0016
nombre de pieces est:0017
nombre de pieces est:0018
nombre de pieces est:0018
nombre de pieces est:0018
nombre de pieces est:0018
nombre de pieces est:0020
Connected 00:27:09 Auto detect 9600 8-N-1 SCROLL CAPS NUM Capture Print echo
```

5. Conclusion

Dans ce chapitre nous avons mis en œuvre le système de comptage de pièces. Ainsi nous avons réalisé les différents montages nécessaires pour la communication RS232, et le détecteur de passage. Nous avons aussi programmé le microcontrôleur pour génère le code binaire du décodeur et piloter en multiplexage l'affichage 7 segments, avec une logique de contrôle simple.





Conclusion

Générale

CONCLUSION GENERALE

Le développement des systèmes à base des microcontrôleurs PIC est devenu de plus en plus courant, ceci est dû à plusieurs causes : beaucoup de ressources internes (périphériques intégrés), mémoires embarquées de plus en plus grande, vitesse de calcul accrue... En effet, il est un composant parfaitement adapté à des applications embarquées, il permet de réaliser des applications diverses qui peuvent servir dans différents domaines industriels ou pour les projets de fin d'étude des étudiants.

Après avoir atteint nos objectifs et réaliser avec succès notre projet, nous pouvons conclure que ce projet nous a énormément appris sur les applications à base de PIC. Nous pouvons dire aussi que c'était une initiation claire sur la programmation des microcontrôleurs autant au niveau de l'électronique que l'algorithmique. Cependant, ce travail reste, comme toute œuvre humaine, incomplet et perfectible, nous recommandons d'en améliorer la conception et pour cela nous proposons des améliorations pour les développements futurs:

- Créer une application terminale pour exploiter les données communiquées par le PIC, exemple calcul du flux des pièces.
- Améliorer le détecteur de passage pour différencier les pièces par exemple
- Tester d'autres modes de communication : RS480, USB....

Les améliorations qui peuvent être apportées à notre travail sont nombreuses et dépendent de l'application souhaitée.

En fin, ce projet a des perspectives pédagogiques et nous espérons qu'il sera très utile pour les étudiants qui étudieront la programmation des microcontrôleurs.



Bibliographie

BIBLIOGRAPHIE

- [1] BALDWIN K., PIEDRA R.M., Overflow Avoidance Methodologies in Commonly Used DSP Functions, Proc. of the International Conference on Signal Processing Applications and Technologies, Toronto, Canada, 1998.
- [2] <http://www.usinenouvelle.com/article/electronica-la-bataille-des-microcontrôleurs-a-basse-consommation-bat-son-plein>.
- [3] LA PROGRAMMATION DES PICS PAR BIGONOFF PREMIERE PARTIE – PIC16F84 – Révision 6
- [4] Programmation des PIC en C Christian tavernier, L'usine nouvelle DUNOD
- [5] <https://fr.m.wikipedia.org/wiki/Compteur>
- [6] PASCAL MAYEUX, Apprendre la programmation des PIC par l'expérimentation et simulation 3^e Édition par technique et scientifiques françaises.
- [7] Document technique Microchip PIC 16f62x DataSheet
- [8] Microcontrôleurs PIC 16f628 (version 1.00 février 2002) par Jacques WEISS, SUPLEC Campus de Rennes.
- [9] Pierre Mayé Optoélectronique industrielle conception et application, DUNOD.
- [10] disciplines.ac-bordeaux.fr/cours/39_s4_5_acquisition_de_donnees.doc
- [11] Advanced PIC microcontroller projects in C-Dogan Ibrahim-newnes 2008.
- [12] <http://www.mikroe.com/mikroc/pic/>
- [13] <https://fr.wikipedia.org/wiki/Proteus>
- [14] datasheetcatalog.net/key/MAX232 ; [15] datasheetcatalog.net/key/74LS48
- [16] <http://www.valise-diagnostique.fr/fr/programmeur-universel/312-programmeur-universel-minipro-tl866cs.html>