République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mouloud MAMMERI, Tizi-Ouzou

Faculté de Génie Electrique et d'Informatique
Département d'Automatique

Mémoire de Fin d'Etudes

En vue de l'obtention du diplôme

D'Ingénieur d'Etat en Automatique

Thème

Commande à distance d'un chariot mobile avec évitement d'obstacle

Proposé et Dirigé par :

Mr MELLAH R.

Présenté par :

SALHI Farid.
REZKI Dahmane.

Soutenu le : / 07 /2010

Promotion 2010

REMERCIEMETS

En premier lieu nous remercions le bon DIEU le tout puissant et le créateur de nous avoir facilité le chemin et qui nous a donné durant toutes ces années la santé, le courage à la fois en nous même pour arriver à ce jour.

Nous tenons à remercie, notre promoteur Mr MELLAH Rabah de nous avoir encadré, suivi et orienté tout au long de ce projet.

Nos remerciements s'adressent également aux enseignants qui ont contribué à notre formation, ainsi que les membres du jury qui nous ferons l'honneur de juger notre travail.

Nous remercions enfin, la responsable du laboratoire de mesure pour son aide, et tous ceux qui de près ou de loin ont contribué de manière significative à l'élaboration de ce mémoire.

Dédicaces

Je dédie ce modeste travail à :

Mes très chers parents pour leur bienveillance, leur affection et leur soutien au long de ma carrière scolaire et universitaire.

A la mémoire de ma grand-mère

A mes chers frères et leurs femmes

A ma sœur, et son mari

A ma tante, et sa famille

A tous mes cousins et cousines

A tous mes oncles

A toute ma famille

A toute la promotion Automatique 2010

Farid.

Dédicaces

Je dédie ce modeste travail à :

Mes très chers parents.

A la mémoire de mon grand-père

A mes grande mères Zhira et Fatma.

A mon grand-père.

A mes frères et sœurs.

A toute ma famille.

A tous les amis et camarades sans exception.

A toute la promotion Automatique 2010

DAHMANE.

SOMMAIRE

CHAPITR II : Unité de commande

CHAPITRE III Unité de perception de l'environnement

CHAPITRE IV Télécommande de pilotage

CHAPITRE V La carte de commande

Introduction générale

La robotique est le domaine où se conjuguent plusieurs spécialités en vue de réaliser une entité pourvue de possibilités d'interagir avec son environnement, par une intervention minimale d'un operateur humain. Selon l'importance de la sollicitation en question, le robot acquiert un degré spécifique d'autonomie. Il existe différentes catégories de robots, et celle qui a fait l'objet du présent travail est la robotique mobile.

Les robots mobiles prennent actuellement une place importante dans la vie quotidienne du grand public. Notamment dans le domaine de la sécurité civile divers prototypes, chargés de pénétrer dans des endroits hostiles en cas d'incendie, de tremblement de terre etc…. De plus, d'autres applications comme la conquête spatiale fait intervenir principalement des robots mobiles dont la mission opportunité est la dernière en date. Néanmoins, toutes ces applications ne pouvaient pas être mises en pratique sans avoir associe aux robots mobiles le concept de la commande à distance.

Le contexte dans lequel s'inscrit notre travail, consiste à concevoir et réaliser un système de commande automatisé à distance d'un chariot mobile en tenant compte de l'évitement d'obstacles. La commande à distance s'effectue via une télécommande infrarouge à base de deux microcontrôleurs, dont l'un en émission et l'autre en réception. Celui de l'émission permet de générer le code approprié à une touche appuyée en utilisant le code RC5, et le transmettre par la suite au récepteur à travers une diode infrarouge. Quant à celui du récepteur a pour objectif de décoder le message reçue, par le récepteur infrarouge TSOP 1738 et d'exécuter la tâche appropriée, permettant ainsi au chariot d'effectuer l'un des mouvements à savoir, rouler en marche avant et en arrière selon l'orientation à gauche ou à droite, tout en assurant les fins de course.

Pour permettre au chariot d'éviter les obstacles, nous avons doté notre système de commande d'un capteur à ultrason qui permet de calculer la distance sans contact avec les obstacles. Ce capteur à ultrason est entrainé par un moteur pas à pas de telle manière, dés que le chariot se rapproche d'un obstacle, il s'arrête, tester la voie libre à suivre (gauche ou droite) afin de contourner l'obstacle.

Pour gérer l'ensemble des circuits de commande, nous avons utilisé une unité de traitement puissante, un microcontrôleur PIC 16F877A, qui est programmé pour atteindre le but de la réalisation (le téléguidage par télécommande et l'évitement d'obstacles).

Ce mémoire est structuré de la façon suivante:

Le premier chapitre présent les notions de bases de la robotique mobile ainsi que les différentes méthodes de perception de l'environnement, par la suite on va faire une description de la structure mécanique du notre robot.

Le deuxième chapitre est consacré à la description de l'unité de base (PIC 16F877A) avec laquelle on a géré notre système de commande.

Le troisième chapitre fait l'objet de l'étude appropriée à l'unité de perception de l'environnement (les capteurs à ultrasons).

Le quatrième chapitre présent la conception matérielle et logicielle de la télécommande infrarouge de pilotage du robot à distance.

Le cinquième chapitre décrit la conception matérielle et logicielle de la carte de commande du robot.

En fin, nous terminons notre travail par une conclusion générale et quelques perspectives envisagées.

CHAPITRE I

LES ROBOTS MOBILES

I.1 Introduction

Depuis les années 70 les chercheurs sont intéressés à l'étude d'un nouveau type de machine appelé robot. Cette nouvelle machine se caractérise par sa capacité à être programmée pour réaliser des tâches très diverses. Ses capacités en matière de manipulation d'objets lui ont permis de s'intégrer dans des lignes de production industrielle, où l'environnement difficilement supportable par l'homme (condition externe de température ou de pression, radioactivité, etc..).

Nous définissons un Robot Mobile Autonome (RMA) comme étant un système programmé, disposant de moyens de traitement de l'information permettant une capacité décisionnelle suffisante et de moyens matériels adaptés (la charge utile), de façon à pouvoir exécuter, sous contrôle humain réduit, un certain nombre de tâches précises, dans un environnement variable, non complètement connu à l'avance.

Les domaines d'application des robots mobiles sont très variés : le transport de courrier et de marchandises; l'exploration en environnement hostile, par exemple le milieu nucléaire, l'espace et les milieux sous-marin; les chantiers de construction : le déminage, aussi bien dans le domaine civil tels que les aéroports que dans le domaine militaire; la lutte contre les incendies; l'assistance aux personnes handicapées ou malades; etc.… Il existe même des robots mobiles de compagnie.

Par leurs caractéristiques, les robots mobiles ont fait l'objet de différentes études. Celles-ci sont motivées par les nombreux problèmes que la robotique mobile doit résoudre pour assurer son rôle. Des problèmes tels que la conception mécanique, la perception de l'environnement, la modélisation de l'espace de travail, la planification de trajectoires sans collision, les lois de contrôle non linéaires, sont des différents sujets de recherche de la robotique mobile.

I.2 Conception d'un robot mobile autonome

Un robot mobile autonome comporte trois sous-systèmes comme le montre la figure I-1 :

- La structure mécanique.
- La charge utile.
- La structure de commande.

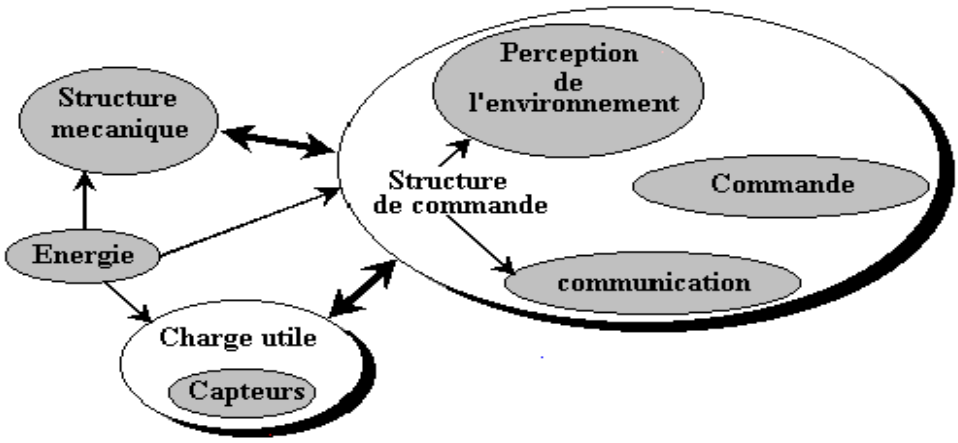


Figure I-1 : Décomposition d'un Robot mobile autonome

I.2.1 La Structure mécanique

C'est l'ensemble des éléments mécaniques et électriques permettant d'assurer la fonction de mobilité. Nous avons retenus quatre mécanismes de locomotion les plus utilisés :

I.2.1.1 Robots mobiles à roues

Compte tenu de la simplicité du mécanisme de locomotion, ce type de robot est le plus répandu actuellement. La plupart des robots mobiles à roues opèrent dans des sites aménagés: des sites industriels ou des environnements intérieurs, mais il existe également des applications en environnement extérieur, comme l'exploration spatiale. La grande majorité des robots de ce type présentent des contraintes de non holonomie qui limitent le mouvement instantané que le robot peut réaliser. Ces contraintes ont pour conséquence d'augmenter la complexité du problème de planification de trajectoire et son contrôle. La figure I.2 montre quelques types de robots mobiles à roues.


Figure I-2 *:* exemples de robots mobiles à roues

I.2.1.2 Robots mobiles à chenilles

Lorsque le terrain est accidenté, les roues perdent leur efficacité de locomotion. Ceci limite la capacité d'évolution du robot mobile équipé de ce type de système de locomotion. Dans ces conditions, les chenilles sont plus intéressantes, car elles permettent d'augmenter l'adhérence au sol et de franchir des obstacles plus importants. Ce type de robots présente également des contraintes de non holonomie.


Figure I. 3 : robot mobile à chenilles

I.2.1.3 Robots mobiles à pattes

Dans la situation où le terrain est encore plus incertain, avec des grandes différences de hauteur comme par exemple un escalier ou un terrain très accidenté, les robots à roues ou à chenilles ne sont plus efficaces. Pour surmonter ces problèmes les chercheurs en Robotique mobile ont fait recours aux robots mobiles à pattes ayant des points d'appui discrets sur le terrain. Néanmoins, la conception et le contrôle d'un engin à pattes est très complexe. En plus, sa vitesse d'évolution est généralement très réduite. La figure I.4 montre deux types de robot mobile à pattes.

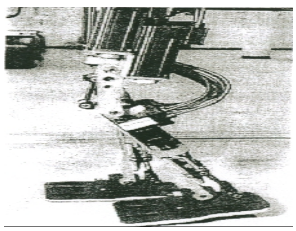



Figure I.4 : (a) : le robot BIP2000,LMS (b) : Honda,Japan

I.2.1.4 Autres moyens de locomotion

Il est d'usage de mettre dans ce groupe tous les robots mobiles qui utilisent un moyen de locomotion différent des trois précédents. Par exemple, les robots mobiles qui se déplacent par reptation, les robots sous-marins, les robots pour l'exploration spatiale, les robots volants (également appelés « drones »), etc. Les applications de ce type de robots sont très spécialisées et les architectures des robots sont en général spécifiques à l'application visée.

I.2.2 La charge utile

La mobilité du robot n'est pas une fin en soi. Le déplacement est dicté par une action à réaliser sur l'environnement. La charge utile concerne directement l'application du robot.

I.2.3 La structure de commande

La structure de commande repose sur trois modules fonctionnant de manière indépendante entre eux. Il s'agit des modules :

- Perception de l'environnement.
- Communication Homme-machine.
- Commande.

I.3 Système de Localisation

La localisation instantanée est l'un des points les plus importants et les plus délicats des robots mobiles, car elle permet de définir le positionnement soit du mobile dans l'environnement ou d'élément particulier de l'environnement.

I.3.1 La localisation du mobile

La localisation du mobile consiste à définir la position en termes de coordonnées d'un point du mobile par rapport à un référentiel de base. Les techniques les plus utilisées sont de deux types:

I.3.1.1 Localisation relative

La localisation relative consiste à déterminer la variation des coordonnées de position lors d'un déplacement. L'estimation de la position absolue est le résultat de l'intégration des déplacements élémentaires. L'inconvénient de cette méthode réside dans l'accumulation des erreurs de mesure et de calcul.

I.3.1.2 Localisation absolue

Les techniques de localisation absolue assurent à la fois la mesure de la position et l'orientation du mobile à tout instant. Il existe de nombreuses méthodes de localisation, dont le choix de la technique est dicté par le type de la tâche à réaliser.

I.3.2 La localisation de l'environnement

Pour cela plusieurs techniques sont employées, on cite :

I.3.2.1 Les méthodes télémétriques

Les télémètres utilisent un émetteur et un récepteur. Selon le rayonnement émis , on distingue les radars, les sonars (ultrasons) ou les télémètres optiques.

La mesure de la distance repose sur deux approches : la mesure du temps de vol ou par la méthode de triangulation.

I.3.2.1.1 Télémétrie directe par mesure du temps de vol

En robotique on utilise généralement les ultrasons, les infrarouges et les télémètres laser.

I.3.2.1.1.1 Télémètre à ultrasons

Les télémètres à ultrasons utilisent des vibrations sonores dont les fréquences ne sont pas perceptibles par l'oreille humaine. Les fréquences couramment utilisées dans ce type de technologie vont de 20 kHz à 20 MHz.

Les ultrasons émis se propagent dans l'air et sont réfléchis partiellement lorsqu'ils heurtent un corps solide, en fonction de son impédance acoustique. L'écho en retour prend la forme d'une onde de pression, à l'image des vaguelettes circulaires déformant la surface de l'eau lorsqu'on jette une pierre. La distance entre la source et la cible peut être déterminée en mesurant le temps de vol séparent l'émission des ultrasons et du retour de l'écho.

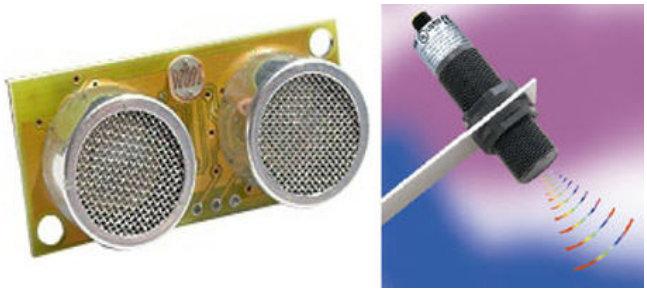


Figure I-5 : Télémètres ultrasonores MSU08 et Migatron RPS 409 IS

I.3.2.1.1.2 Télémètre à infrarouge

Les capteurs infrarouges sont constitués d'un ensemble émetteur/récepteur fonctionnant avec des radiations non visibles, dont la longueur d'onde est juste inférieure à celle du rouge visible. La mesure des radiations infrarouges étant limitée et en tout état de cause, la qualité très dégradée au delà d'un mètre. Ces dispositifs ne servent que rarement de la télémétrie et on les rencontre plus souvent comme détecteurs de proximité.

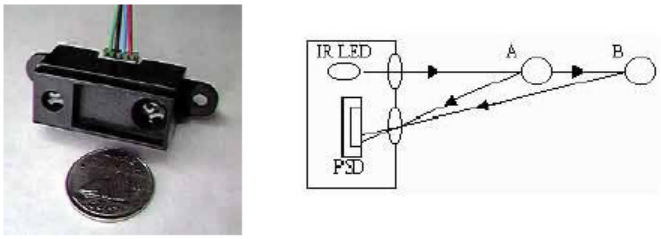


Figure I-7 : Télémètres infrarouges Sharp

I.3.2.1.1.3 Télémètre laser

Le laser est beaucoup utilisé en télémétrie car il permet de créer des faisceaux monochromatiques de grande directivité. Il est possible de localiser des objets situés à quelques dizaines de mètres avec précision de moins d'un centimètre. Son principe de fonctionnement est le suivant :

A un instant donné, une impulsion lumineuse très courte est envoyée par l'intermédiaire d'une diode laser de faible puissance, la réflexion de cette onde donne un écho qui est détecté au bout d'un temps proportionnel à la distance capteur - obstacle.

La direction des impulsions est modifiée par rotation d'un miroir, l'angle de balayage couvrant généralement entre 100 et 180 degrés sur des produits commerciaux.

Figure I-8 : La famille de télémètre laser Sick.

I.3.2.1.2 Télémétrie par triangulation

Le principe consiste à mesurer l'angle entre la direction d'un faisceau émis et celle sous laquelle est vu le point d'impact sur la cible. On utilise une source active permettant de projeter un point, une ligne ou même une grille.

I.3.2.2 L'analyse d'image

La détermination des paramètres particuliers s'effectue par l'analyse d'une ou plusieurs images. La stéréoscopie notamment assure la détermination de l'information de profondeur (ou distance) de l'environnement. Les caractéristiques des formes (ex: droites représentant les bords d'une route) de couleur ou texture sont également employées. Ces informations assurent la détermination locale de l'environnement.

I.3.2.3 La localisation par contact

La localisation par contact entre un élément physique lié au mobile et l'environnement constitue une technique analogue à la démarche de l'aveugle. Le principe consiste à tendre un bras télescopique vers l'environnement. La relève de l'état du bras lors d'un contact définit la position de l'environnement par rapport au mobile.

I.4 La Sécurité en Robotique

Un robot, selon la tâche qui lui est confiée, peut être amener à travailler au voisinage du personnel. A ce titre, il est obligatoire qu'il soit doté d'organes garantissant la sécurité. Des capteurs sont disponibles tout autour d'un mobile afin de détecter un obstacle sur le domaine le plus étendu possible. Deux types de capteurs sont employés :

- Les capteurs de proximité assurent la détection avant collision (ultrason, hyperfréquence, infrarouge, …).
- Les capteurs à contact détectent une collision ou un choc avec l'environnement (contact électrique sur pare-chocs, résistance variable, fibre optique.).

I.5 la structure mécanique du robot sur lequel on travail

Comme nous avons indiqué au début, il y a trois structures mécaniques des robots mobiles (mobiles à roues, mobiles à pattes et mobiles à chenille). Le but de notre travail est de réaliser un modèle mécanique du robot prêt à être commandé électriquement par un système facile à réaliser et à gérer. Le Robot sur lequel on a travaillé, est un chariot mobile à quatre roues, deux roues motrices en arrière et les deux autres directrices en avant.

I.5.1 Caractéristiques du robot (chariot)

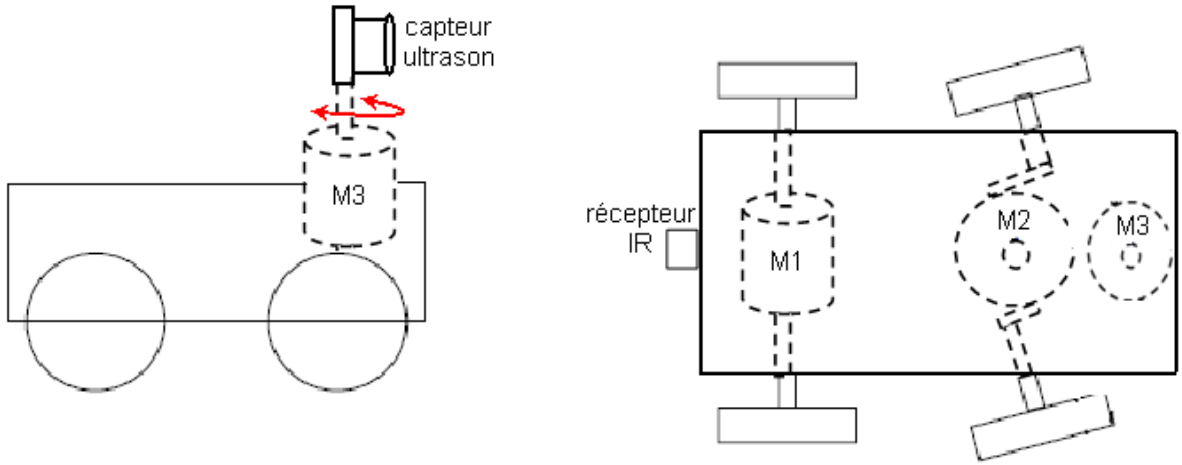


Figure I.9 : la structure du chariot

M1 : moteur à courant continu, positionné horizontalement

(En parallèle avec l'axe des roues arrière). Sert à propulser le robot en avant ou en arrière.

M2 : moteur à courant continu, positionné verticalement (Perpendiculaire avec l'axe des roues de devant). Sert à assurer la rotation du robot à gauche ou à droite.

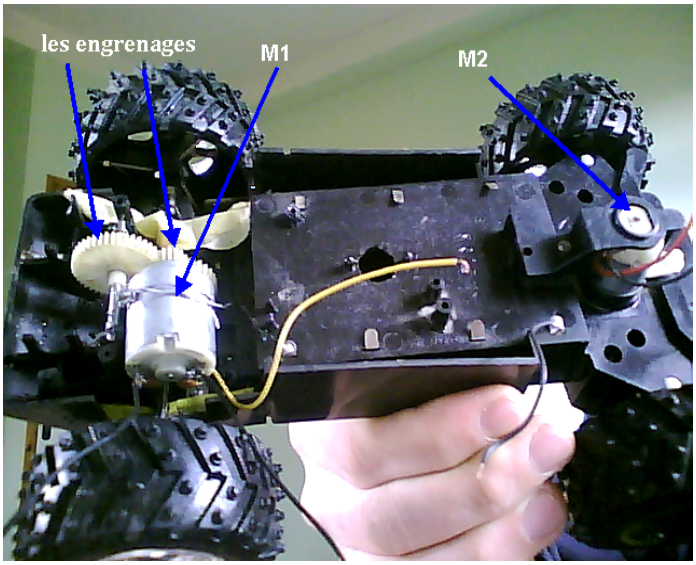


Figure I.10 : vue de dessus du chariot

M3 : moteur pas à pas bipolaire, positionné verticalement. Ce moteur est surmonté d'un capteur à ultrason (Sous forme d'une tourelle), comme le montre la figure ci-contre :

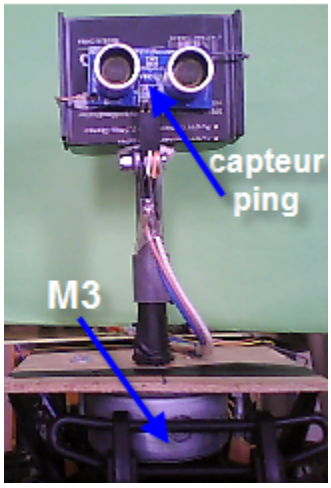


Figure I.11 : la tourelle

I.5.2 Types des moteurs utilisés

Nous avons utilisé deux moteurs à courant continu et un moteur pas à pas bipolaire :

Le premier moteur à courant continu entraîne le mouvement en avant et en arrière du chariot; et le deuxième est utilisé pour l'orientation du chariot. Quant au moteur pas à pas bipolaire surmonté d'un capteur à ultrason afin de prendre les mesures (distances des obstacles) dans différentes directions.

I.5.2.1 Moteur à courant continu

Le moteur à courant continu est l'actionneur électrique le plus classique, il est toujours présent dans nombreuses applications, il existe beaucoup de structures mais le principe de base est le même avec en particulier le rôle fondamental du collecteur et des balais.

- **Constitution**

Comme toute machine tournante, le moteur à courant continu comporte une partie fixe (le stator) et une partie mobile (le rotor) séparées par un entrefer. Le stator dit aussi inducteur, porte des aimants qui sont chargés de créer le champ magnétique dans l'entrefer, il comporte un pôle nord et un pôle sud. les deux demi culasses permettent de canaliser les lignes de champ. Le rotor porte un bobinage appelé induit. La connexion avec le générateur est faite par des contacts mobiles, qui se composent des balais solidaires du stator et frottent le collecteur lié au rotor.

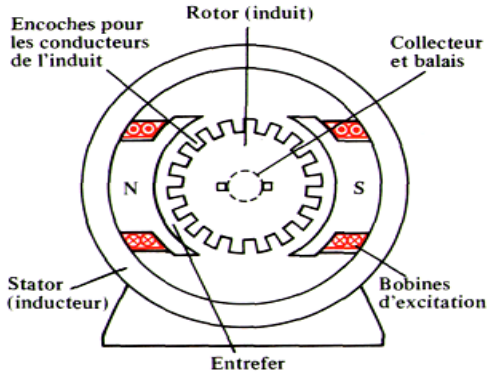


Figure I.12 : Constitution d'un Moteur à courant continu

I.5.2.2 Les moteurs pas à pas

Les moteurs pas à pas, également connus sous le nom de « stepping motor », peuvent tourner et s'arrêter avec une précision de l'ordre du centième de millimètre. Cette très grande précision et leur fiabilité les prédestinent à être utilisés dans de nombreux appareils électroniques comme par exemple, les lecteurs de disquettes pour la recherche des pistes, les imprimantes, les photocopieuses pour l'agrandissement, ainsi dans différents robots industriels.

Un moteur pas à pas, est une machine tournante dont le rotor se déplace d'un angle élémentaire α_p , appelé "pas ", chaque fois que son circuit de commande effectue une commutation de courant dans un ou plusieurs de ses enroulements. Il s'agit donc, avant tout, d'un actionneur de positionnement. Toutefois, une succession rapprochée de commutations permet d'obtenir une rotation continue.

Compte tenu de son principe, la commande de la position ou de la vitesse d'un moteur pas à pas peut se faire **sans asservissement :** il n'est pas nécessaire de contrôler le résultat qui correspond exactement aux ordres donnés à condition de respecter certaines limites de fonctionnement. Ce mouvement par pas est appelé mouvement incrémental. Pour avoir une bonne résolution dans le positionnement la machine doit avoir un pas assez faible, c'est un paramètre essentiel de la machine .on peut également caractériser cette résolution par le nombre de pas par tour " **N** ", qui doit évidement être élevé : $\mathbf{N=2\pi/\alpha_p}$

I.5.2.2.1 Constitution et principe de fonctionnement

Tous les moteurs pas à pas comprennent un stator portant des bobines dans les quelles le courant est commuté par l'électronique de commande ; par contre, des différences apparaissent au niveau du rotor, il y a trois types :(rotor à aimant, à reluctance variable, et hybride **:** combinant l'aimant et la reluctance variable).

Le moteur qu'on a utilisé, est de type Bipolaire a rotor à aimant avec deux pôles, et un stator qui comporte quatre enroulements, ses Caractéristiques Techniques sont :

*Tension nominale : 12 Volts.

* Courant nominal : 330 mA.

* Résistance de bobine : 36 Ohms

* Inductance : 37mH.

* Couple moteur bloqué : 1350 g/cm.

* Nombre de pas par tour : 48.

*Degré d'angle (le pas) : 360°/48 = 7.5°.

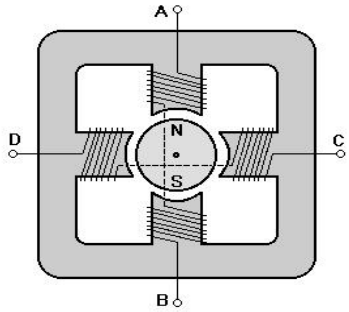


Figure I.13 : Moteur pas à pas bipolaire

I.5.2.2.2 Caractéristiques des moteurs pas à pas Bipolaire

Pour un moteur bipolaire, les enroulements du stator n'ont pas de point milieu comme le moteur pas à pas unipolaire. Chaque borne de chaque enroulement est alimenté soit positivement, soit négativement, d'où la signification du terme : "bipolaire"

En l'absence de courant dans les enroulements, le rotor se place dans une position d'équilibre stable, sa paire de pôles étant en face d'une paire de pôles du stator.

Ce phénomène est du à la structure à pôles saillants de la machine. Pour écarter le rotor de sa position de repos, il faut exercer un couple appelé couple de détente.

Le nombre de phases est égal au nombre d'enroulements.

Le sens de rotation d'un moteur pas à pas bipolaire dépend du sens du courant et de l'ordre d'alimentation des bobinages.

I.6 Conclusion

L'étude des robots mobiles à roues est un domaine de recherche en pleine expansion, et qui présente des enjeux industriels considérables.

Dans ce chapitre, nous avons présenté les notions de bases de la robotique mobile ainsi quelques méthodes de perception de l'environnement. Par la suite, nous avons décrit la structure mécanique de notre robot en tenant compte des actionneurs utilisés pour gérer les différents mouvements.

CHAPITRE II

LE ROBOT RÉALISÉ

Unité De Commande

II.1 Introduction

La partie commande de ce projet est basée sur le microcontrôleur PIC16f877A. En raison de l'utilité de ce composant, nous avons jugé nécessaire de consacré un chapitre pour décrire ce microcontrôleur.

La connaissance de différentes caractéristiques de ce composant va nous aider à mieux concevoir la carte de commande de notre robot.

II.2 Généralité sur les PIC

Un microcontrôleur est une unité de traitement de l'information de type microprocesseur à la quelle on ajoute des périphériques internes permettant de réaliser des montages sans l'ajout de composants externes. En ce sens, les pics sont particulièrement bien dotés, car ils intègrent une mémoire de programme, mémoire de données, ports d'entrée-sortie et même horloge, bien que des bases de temps externes puissent être employées.

Jusqu'à une certaine époque, les microcontrôleurs respectent l'architecteur de Von Neumann. Néanmoins, cette dernière présente quelques inconvénients qui la condamnent comme outil général. En effet, la vitesse d'exécution est limitée, ainsi que les instructions et les données transitent par le même bus qu'utilise simultanément, la mémoire de programme et la mémoire de données comme le montre la figure II.1.

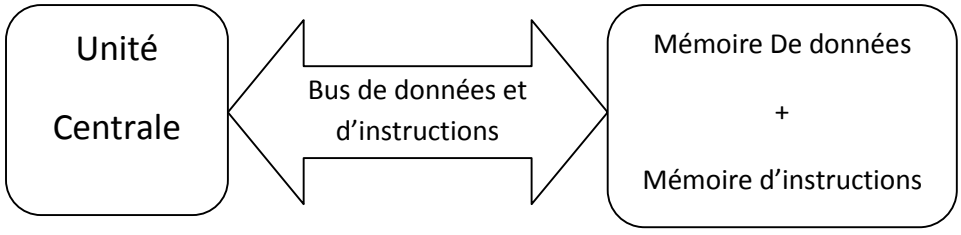


Figure II.1 : Principe de l'architecture Von-Neumann

En revanche, les pics ont été conçus sur une architecture appelée HARVARD dans laquelle les instructions et les données sont clairement différenciés et sont véhiculées sur des bus différents comme le montre la figure II.2.

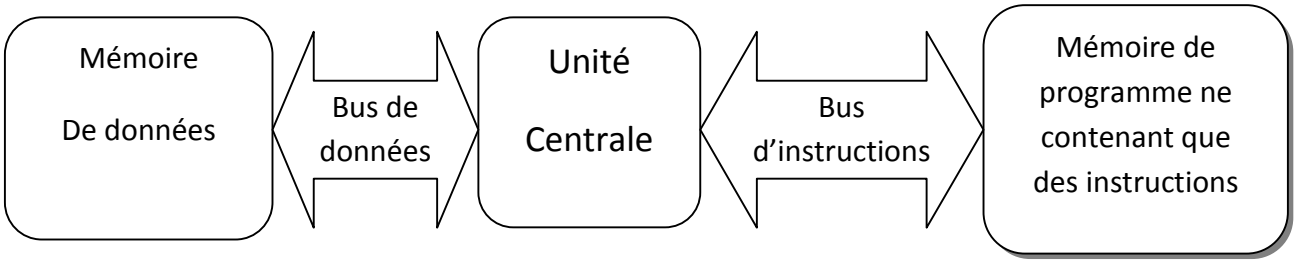


Figure II.2 : Principe de l'architecture Harvard

Par ailleurs, les pics font appel aussi à une architecture de type RISC (reduce instruction set computer) qui signifie : jeu d'instruction réduit. Ça veut dire que les instructions sont codées sur un seul mot, ce qui présente deux avantages qui sont :

- Tous les emplacements de la mémoire de programme contiennent une instruction.
- Un seul cycle machine suffit pour lire le code complet d'une instruction.

D'où un gain de vitesse d'exécution.

Les circuits RISC utilisent ensuite une structure de type Pipe-line qui leur permet d'exécuter une instruction tout en recherchant la suivante en mémoire, là encore accroissement de la vitesse d'exécution.

L'inconvénient majeur de la structure RISC est le jeu d'instruction qui est très pauvre et l'accès aux registres internes et la mémoire, est très délicat.

II.2.1 Identification d'un PIC

La dénomination PIC est sous copyright de MICROCHIP, les autres fabricants sont dans l'impossibilité d'utiliser ce terme. Un pic est identifié de la forme :

NN(L) XXYY-ZZ dont :

NN : famille du composant, actuellement (12, 16,17et18), selon la famille du PIC, il existe trois grande familles selon la taille des instructions :

1) La famille Base-line : ses instructions sont codées sur 12 bits.
2) La famille Mide-range: utilise des instructions codées sur 14 bits.
3) La famille High-end : utilise des instructions codées sur 16 bits.

L : indique que le PIC peut fonctionner avec une plage de tension beaucoup plus tolérante.

XX : indique le type de la mémoire programme, à savoir :

-C : la mémoire programme est une EPROM, ou EEPROM.

-CR : la mémoire programme est une ROM.

-F : la mémoire programme est une mémoire FLASH.

YY : identificateur.

ZZ : indique la fréquence maximale que peut recevoir le PIC.

II.3 Etude du PIC 16F877A

II.3.1 Description

-Consommation : moins de 2mA sous 5V à 4MHz.

-Architecture RISC : 35 instructions de durée 1 ou 2cycles.

-Durée d'un cycle : la période de quartz divisée par 4 (soit 1µS pour un quartz de 4MHz).

-2 bus distincts pour le code programme et les données.

-code d'instruction : mot de 14 bits et un compteur programme (PC) sur 13 bits, ce qui permet d'adresser 8K mots (de '0000'H à '1FFF'H).

-Bus de données sur 8 bits.

-2 compteurs 8 bits et 1 compteur 16 bits avec pré diviseur programmable.

-convertisseur analogique numérique sur 10 bits.

- 2 modules pour la PWM avec une résolution de 10 bits.

-interface avec un autre micro : 8 bits+3 bits de contrôle pour R/W et CS.

-368 octets d'EEPROM de données.

-8K mots de 14 bits en EEPROM Flash pour le programme (h'0000' à h'1FFF').

-1registre de travail : W, et un registre fichier : F, permettant d'accéder à la RAM ou aux registre internes du PIC ; tous les deux sont des registres à 8 bits.

-PORTA : 6 entrée-sorties.

-PORTB : 8 entrée-sorties.

-PORTC : 8 entrée-sorties.

-PORTD : 8 entrée-sorties.

-PORTE : 3 entrée-sorties.

II.3.2 Brochage du PIC 16F877A :

Le brochage du PIC 16f877A est présenté par la figure II.3 :

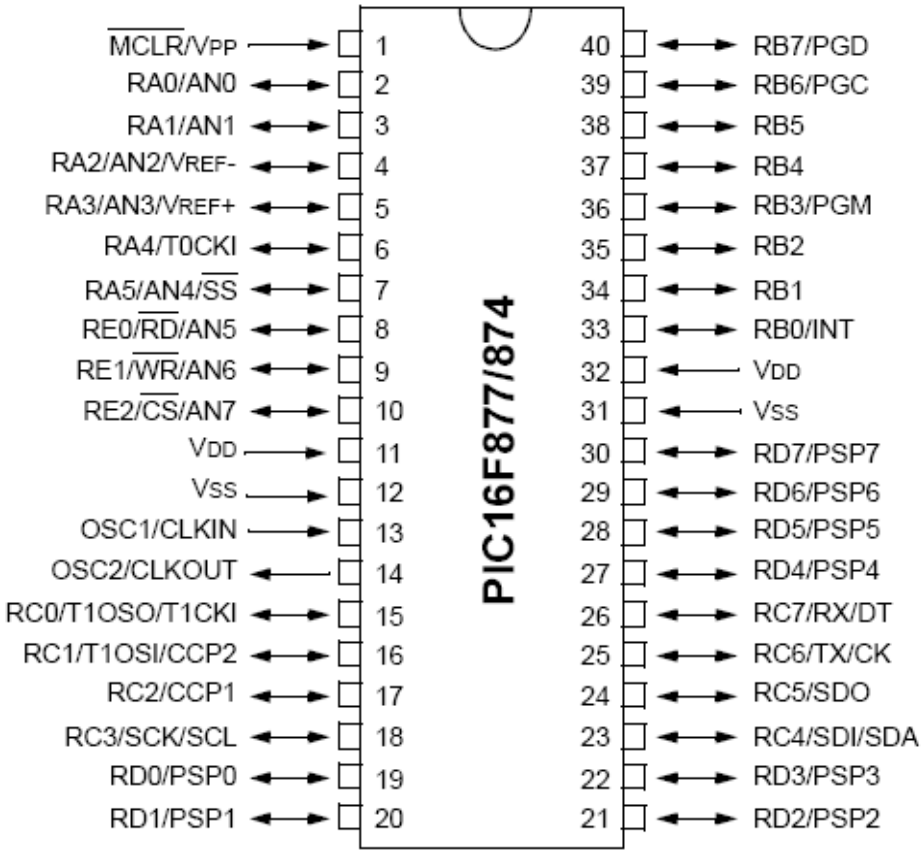


Figure II.3 : Brochage du PIC 16F877

II.3.2.1 Les Fonction des pattes :

- ✓ MCLR/Vpp : entrée du reset manuel ou la tension de programmation.
- ✓ RA0 à RA7 : sont les entées/sorties du port A (bidirectionnel).
- ✓ AN0 à AN7 : sont les entrées analogiques.
- ✓ Vref- et Vref+ : entrées des tensions de référence pour le convertisseur analogique.
- ✓ RE0 à RE2 : entrées/sorties du port E (bidirectionnel).
- ✓ TOCKI : entrée d'horloge externe du timer0.
- ✓ /RD : entrée, lecture en mode PSP.
- ✓ /WR : entrée, écriture en mode PSP.
- ✓ /CS : entrée, chip select en mode PSP.
- ✓ VDD : alimentation positive du circuit.
- ✓ VSS : est la masse du circuit.
- ✓ OSC1/CLKIN et OSC2/CLKOUT : entrée/sortie d'oscillateur d'horloge.
- ✓ RC0 à RC7 : entrée/sorties du port C (bidirectionnel).
- ✓ T1OSI et T1OSO : sont l'entrée et sortie de l'oscillateur du Timer1 respectivement.
- ✓ T1CLKI : entée d'horloge externe pour le Timer1.
- ✓ SCK : E/S d'horloge en mode SPI.
- ✓ SCL : E/S d'horloge pour le mode I2C.
- ✓ CCP1 et CCP2 : E/S pour les modes (compare, capture et pwm).
- ✓ RD0 à RD7 : E/S du port D (bidirectionnel),
- ✓ PSP0 à PSP7 : entrées /sorties du port parallèle PSP.
- ✓ RB0 à RB7 : entrées/sorties du port B (bidirectionnel).
- ✓ SDI et SDO : entrées et sorties de donnée en mode SPI.
- ✓ SDA : entrée/sortie de donnée pour le bus I2C.
- ✓ INT : entrée d'interruption externe.
- ✓ RX : est une entrée, pour la réception asynchrone SCI.
- ✓ TX : sortie, pour l'émission asynchrone SCI.
- ✓ DT : entrée/sortie de données synchrone SCI.
- ✓ CK : entrée/sortie de l'horloge synchrone.

II.3.3 Architecture interne :

Les éléments constitutifs de l'organisation interne du PIC16F877A sont :

- Une unité arithmétique et logique (ALU).
- Un système de génération d'horloge à partir du quartz externe (timing generation).
- Une pile de 8 niveaux (stack).
- Un compteur programme PC (program counter).
- Registre d'instruction, il contient le code de l'instruction à exécuter.
- Un système d'initialisation à la mise sous tension (power up-timer, power on reset, low voltage programming……).
- Un chien de garde (watchdog).
- Un registre d'état.
- Un registre de travail W, qui est utilisé pour l'adressage.
- Un bus de données (data bus).
- Un bus pour le programme (program bus).
- Registres spéciaux (FSR), représente l'ensemble des registres de configuration.

Les autres éléments sont déclarés déjà au niveau des caractéristiques du PIC.

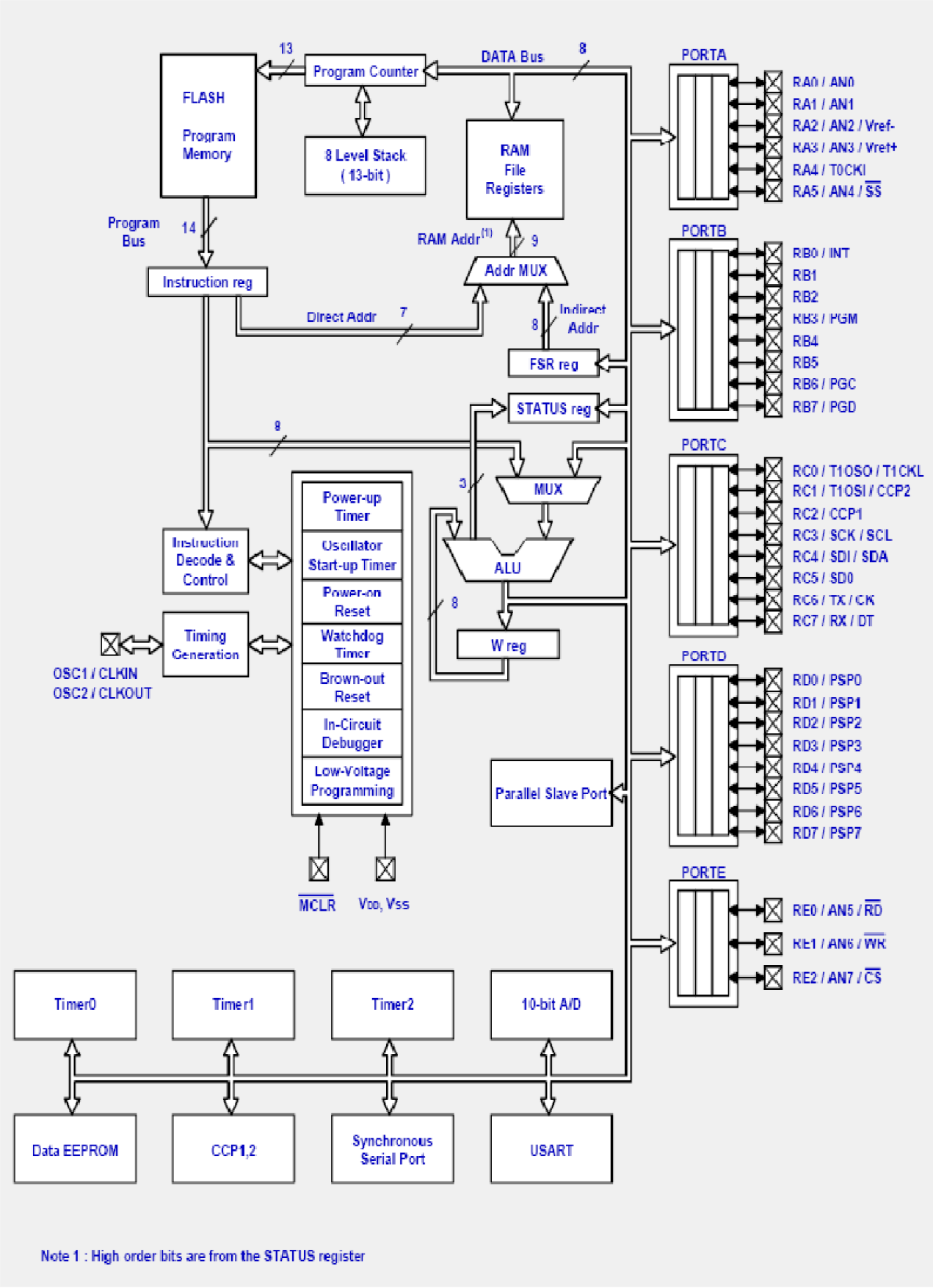


Figure2.3 : Architecture interne du PIC 16F877A

II.3.4 Organisation de la mémoire :

Le PIC 16F877A contient trois blocs de mémoire : la mémoire de données, la mémoire programme qui sont adressables et un espace mémoire non adressable contient la pile.

II.3.4.1 La mémoire de données

II.3.4.1.1 La RAM

C'est une mémoire de travail (volatile), son contenu sera perdu à chaque coupure d'alimentation.

Elle est partitionnée sur quatre Banks, ces dernières contiennent :

- Des registre (GPR) ou à usage générale : ils sont utilisés comme mémoire vive de travail, afin de stocker des variables.
- Des registres spéciaux (SFR) : ils ont des fonctions bien spécifiques allouées à un périphérique donné.

L'ensemble des registres est donné par l'architecture suivante :

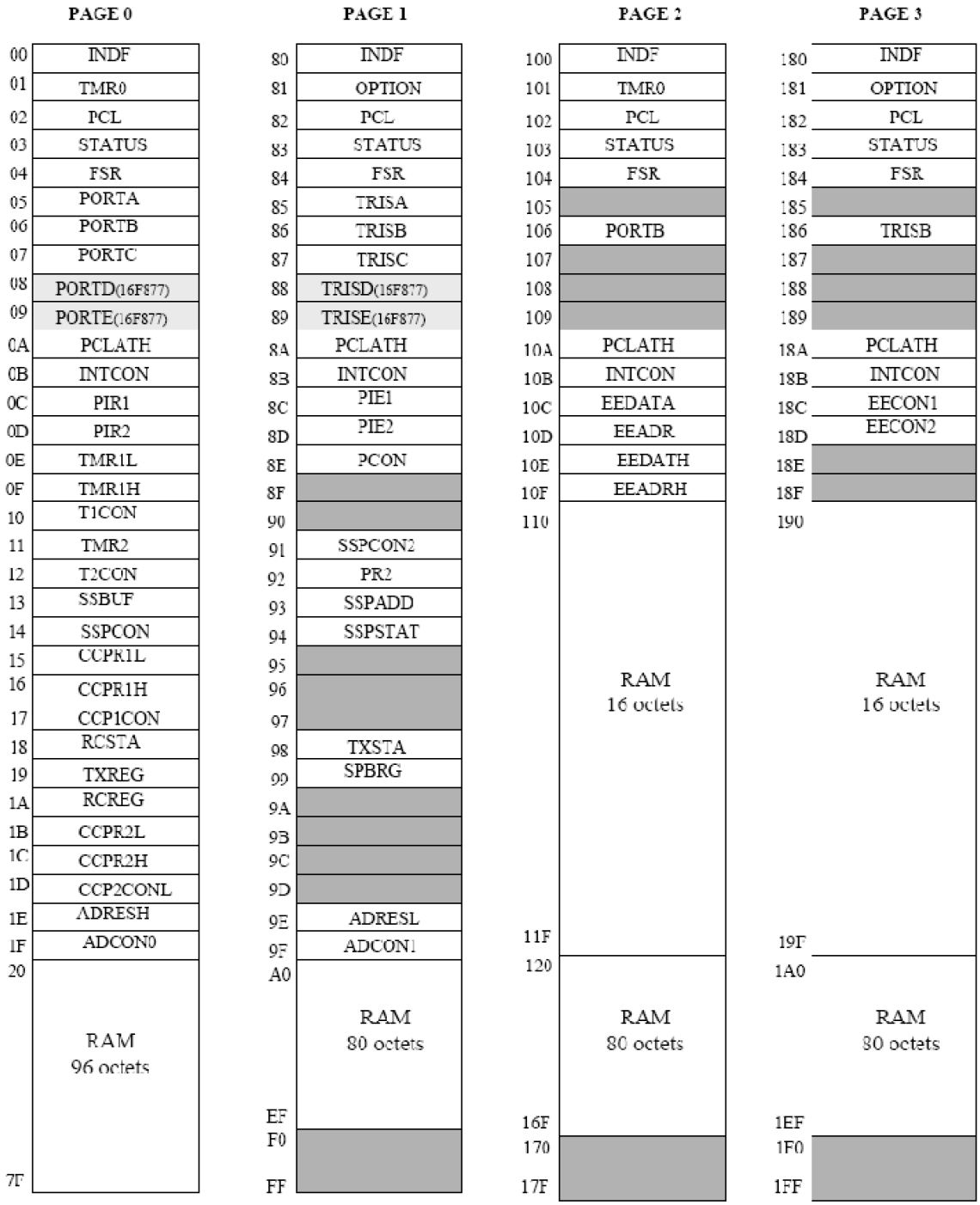


Figure 2.3 : la mémoire de données : RAM avec ses différents Banks.

II.3.4.1.2 L'EEPROM

L'EEPROM est plutôt une mémoire de stockage de données à long terme, qui est constituée de 256 octets que nous pouvons lire et écrire depuis notre programme. Elle est utilisée pour conserver des paramètres semi-permanents.

II.3.4.1.3 La mémoire programme

La mémoire programme est de type FLASH de capacité : 8K mots de 14bits sert à stocker le programme. Après compilation de programme (code), le compilateur génère un fichier «.hex». Celui-ci est transféré ensuite dans la mémoire programme du PIC à l'aide d'un programmateur de PIC.

Cette mémoire n'est pas reliée au bus de données (DATA BUS). Ainsi, sa vocation est de stocker le programme de PIC, mais pas les variables de programme.

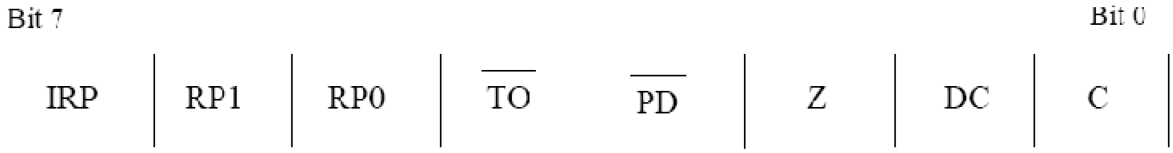
II.3.4.1.4 La Pile (stack)

Le PC (Program conter) est le compteur qui pointe dans la mémoire programme la prochaine instruction à exécuter, Il est lié à la pile du PIC 16F877A qui possède 8 niveaux. Cela signifié qu'on peut avoir jusqu'à 8 imbrications des sous-programmes (interruptions ou fonctions).

II.3.4.2 Les registres internes

Parmi les registres internes du PIC 16F877A, nous allons citer les registres suivant :

II.3.4.2.1 Le registre d'état (Status register)



On accède indifféremment à ce registre par une quelconque de ces 4 adresses.

Au reset : STATUS=00011XXX

IRP : permet la sélection des pages en adressage indirect comme suit :

IRP=0 pour la page 0 (de 00à7F) et la page1 (de 80 à FF).

IRP=1 pour la page 2 (de100 à 17F) et la page 4(de 180 à 1FF).

RP1 et RP0 : permettent la sélection des pages en adressage direct conformément ce tableau :

RP1	RP0	Page sélectionnée
0	0	Page 0 de 00 à 7F
0	1	Page 1 de 80 à FF
1	0	Page 2 de 100 à 17F
1	1	Page 3 de180 à 1FF

Le bit/TO (time out bit)

Active au niveau bas. Il est mis à 0 lorsqu'un débordement du timer, chien de garde(WDT) se produit. Il est mis à 1 suite à une mise sous tension ou à l'exécution d'une instruction CLRWDT ou SLEEP.

Le bit/PD (power down)

Ce bit est mis à 0 lors de l'exécution de l'instruction SLEEP. Il est mis à 1 suite à une mise sous tension ou à l'exécution d'une instruction CLRWDT.

Le bit z (zéro)

Ce bit est mis à 1 si le résultat de l'opération arithmétique ou logique exécutée est nul ,et à 0 dans le cas contraire.

Le bit DC (digit carry)

C'est le bit de retenu pour l'arithmétique en BCD.

Il est mis à 1 quand il aura une retenue sur le 4 ième bit des poids faibles, après les instructions : ADDWF et ADDLW. Il est mis à 0 quand on n'a pas de retenue sur le 4 ième bit des poids faibles.

Le bit C (carry)

C'est le bit de retenue en addition. Il est positionné à 1 par les instructions ADDWF et ADDLW si une retenue est générée depuis le bit de poids fort, alors qu'il reste à 0 dans le cas contraire.

II.3.4.2.2 Le registre option register

Ce registre en lecture et écriture permet de configurer les prédiviseurs du timer et du Watchdog, la source du timer, le front des interruptions et le choix du pull up sur le port B.

Bit 7 Bit 0

$\overline{\text{RBPU}}$	INTEDG	TOCS	TOSE	PSA	PS2	PS1	PS0

Au reset : 11111111

- **RBPU:** Active ou désactive les résistances internes de tirage vers le haut (pull-up).
- **INTEDG** : Permet de choisir le sens de déclenchement des interruptions sur la broche RB0/INT :

  INTED= 1 : Interruption si front montant.

  INTED = 0 : Interruption si front descendant.
- **Le bit TOCS :** Détermine la source d'horloge qu'utilise le timer0 :

  TOCS=1 : L'horloge du Timer est l'entrée RA4/TOClk (pin 6).

TOCS=0 : Le Timer utilise l'horloge interne du PIC.

- **Le bit TOSE :** pour la sélection du front déclencheur de l'incrémentation du TIMER0.

  TOSE= 1: Incrémentation sur front montant.

  TOSE= 0: Incrémentation sur front descendant.

- **Le bit PSA :** Affectation du prédiviseur :

  PSA=1 : Le prédiviseur est affecte au watchdog.

  PSA=0 : Le prédiviseur est affecte au timer0.

- **Les bits PS0, PS1 et PS2 :** Permettent le choix du prédiviseur comme le montre le tableau suivant :

PS2 PS1 PS0	Prédiv Timer	Prédiv Watchdog
0 0 0	2	1
0 0 1	4	2
0 1 0	8	4
0 1 1	16	8
1 0 0	32	16
1 0 1	64	32
1 1 0	128	64
1 1 1	256	128

II.3.4.2.3 Le registre INTCON

Il permet d'autoriser les interruptions globales (**GIE**), les interruptions des Périphériques (**PEIE**), L'interruption **TIMER0** (**T0IE**), l'interruption extérieure (INT/RB0) et l'interruption de changement d'état du **PORTB** (**RBIE**).

Bit 7 Bit 0

GIE	PEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF

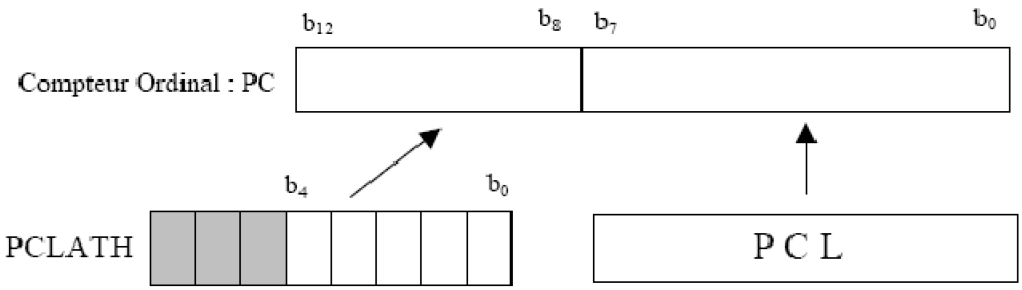
Au reset : INTCON 0000000X

- **GIE :** Validation de toutes les interruptions si (GIE=1). Et interdiction toute interruption si (GIE=0).
- **PEIE :** Validation de l'interruption de fin d'écriture en l'EEPROM (PEIE=1).
- **TOIE :** Validation de l'interruption due au débordement du timer0 (TOIE=1).
- **INTE :** Autorise l'interruption sur la pin RB0/INT (INTE=1).
- **RBIE :** Autorise l'interruption s'il y a changement sur une des entrées RB4 a RB7 si (RBIE=1).
- **TOIF :** Ce flag se positionne a "1" s'il y a débondement du timer0.

- **INTF :** C'est un flag qui se positionne a "1" s'il y a interruption sur la pin 33 (RB0/INT).
- **RBIF :** C'est un flag qui se positionne a "1" s'il y a interruption sur l'une des entrées RB4 a RB7.

II.3.4.2.4 Le registre PCLATH

Le compteur de programme est sur 13 bits. Les 8 bits de poids faible sont dans le registre PCL qui est en lecture /écriture. Les 5 bits de poids forts ne sont pas lisible mais, on peut les écrire indirectement à travers le registre PCLATH.



II.3.4.2.5 Les registre TRISA, TRISB, TRISC, TRISC et TRISE

Servent à orienter chaque patte des ports associés soit en entrée, soit en sortie.

EX : set_tris_a(0x00) : veut dire que le port A est en sortie

II.3.5 Les interruptions

Un certain nombre d'événements sont susceptible de générer des interruptions qui sont:

- ✓ Débordement de timer0.
- ✓ Changement d'état d'une des pattes RB4 à RB7du port B.
- ✓ Comparateur analogique.
- ✓ Lecteur/écriture terminée sur port parallèle.
- ✓ Front sur RB0/ INT.
- ✓ Fin de conversion analogique numérique.
- ✓ Débordement de Timer1.
- ✓ Timer2 a attient la valeur programmée.
- ✓ Capteur/comparaison de Timer1 avec module CCP1.
- ✓ Capteur/comparaison de Timer1 avec module CCP2.
- ✓ Fin d'écriture en EEPROM.
- ✓ Collision sur bus SSP en mode I2C
- ✓ Fin de transmission d'un octet sur l'USART.

II.3.5.1 La séquence de déroulement

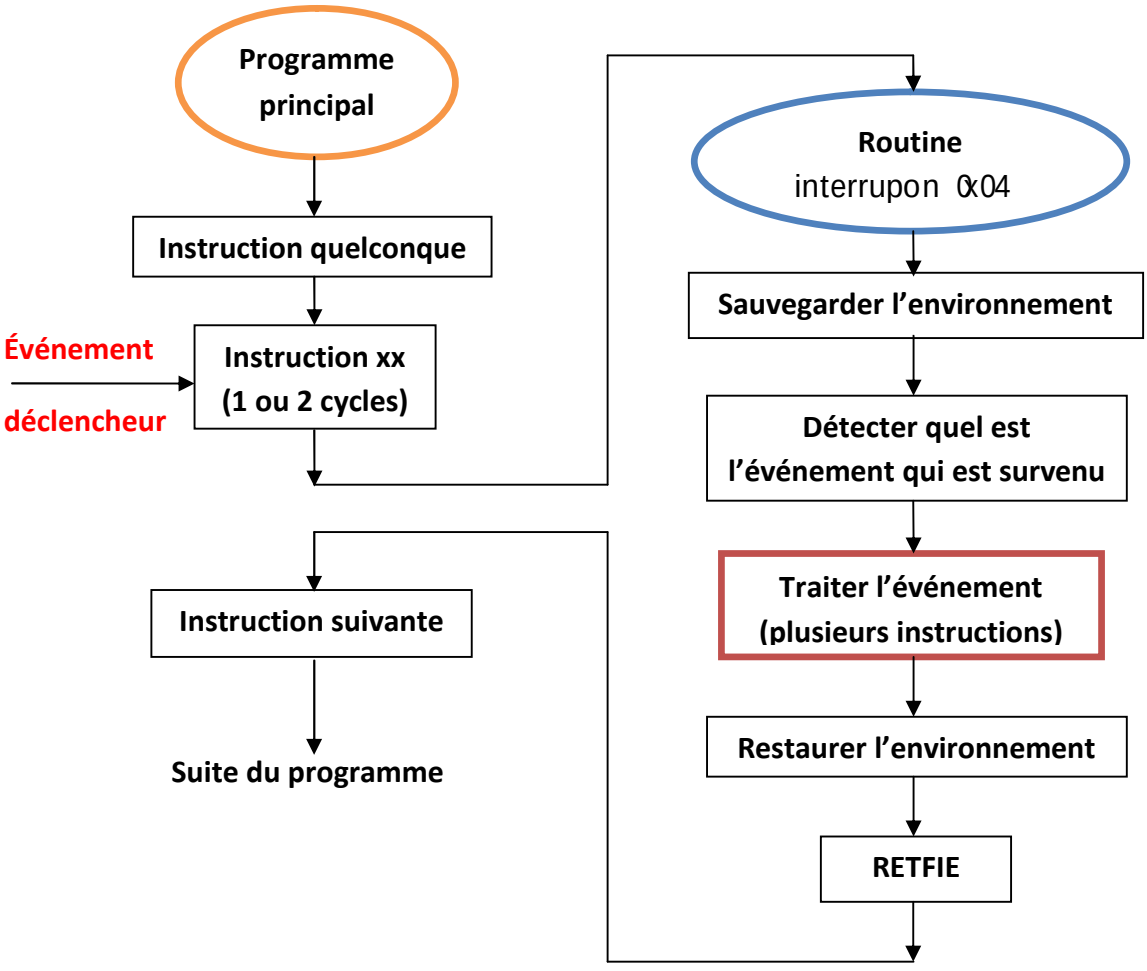
Lorsque l'événement déclencheur d'une interruption intervient, alors son drapeau est positionné à 1 (levé). Si l'interruption a été validée (bit de validation=1), elle est alors déclenchée : le programme arrête ce qu'il est en train de faire et va exécuter la procédure d'interruption qui se trouve à l'adresse 0004 en exécutant les étapes suivantes :

- L'adresse contenue dans le PC (program counter) est sauvegardée dans la pile, puis remplacée par la valeur 0004 (adresse de la routine d'interruption).
- Le bit GIE est placé à 0 pour inhiber toutes les interruptions (afin que le PIC ne soit pas déranger pendant l'exécution de la procédure d'interruption).

A la fin de la procédure d'interruption (instruction RETFIE) :

- Le bit GIE est remis à 1 (autorisant ainsi un autre événement).
- Le contenu du PC est rechargée à partir de la pile ce qui permet au programme de reprendre là ou il s'est arrêté.

II.3.5.2 Organigramme de la séquence de déroulement d'une interruption



FigureII.4: Séquence de déroulement d'une interruption.

II.3.6 Les TIMERs

II.3.6.1 Le TIMER0(RTCC)

C'est un compteur 8 bits (0 à 255) simple, qui compte des impulsions provenant des sources internes ou externes. On peut par ailleurs lui appliquer une prédivision programmable entre 1 et 256.

II.3.6.2 Le TIMER1

Le Timer 1 fonctionne sur le même principe que le Timer0, mais avec un registre de comptage à 16 bits au lieu de 8 bits, ce qui étend notablement ses capacités de comptage. De plus, il possède un mode de fonctionnement particulier qui permet d'utiliser en association avec des modules CCP (module de capture et de comparaison).

II.3.6.3 LeTimer2

Le timer2 a un fonctionnement différent des Tmer0 et Timer1. C'est un compteur à 8 bits avec pré-diviseur et post-diviseur, afin de générer des signaux carrés, en association avec le module CCP (PWM).

Le timer2 est accessible en lecture et écriture, constitue de :

· Un registre de control T2CON (bank0).
· Un prédiviseur (1, 4,16).
· Un registre de période PR2 (bank1) accessible en lecture/écriture.
· Un comparateur,
· Un postdiviseur (1 a 16).

II.3.6.4 Le Timer WATCHDOG

C'est un compteur à 8 bits incrémenté en permanence (même si le µc est en mode sleep) par une horloge RC intégré, indépendante de l'horloge système. Lorsqu'il déborde, (WDT time out), deux situations sont possibles :

- Si le µc est en fonctionnement normal, le « WDT time out » provoque un RESET. Ceci permet d'éviter de rester planté en cas de blocage du microcontrôleur par un processus indésirable non contrôlé.

- Si le µc est en mode SLEEP, le « WDT time out » provoque un WAKE-UP, l'exécution du programme continue normalement là où il s'est arrêté avant de rentrer en mode SLEEP.

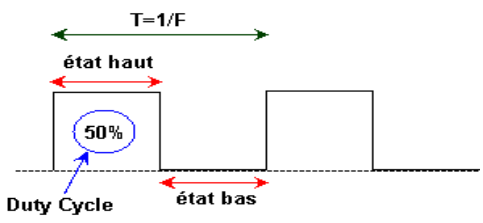
Cette situation est souvent exploitée pour réaliser des temporisations.

L'utilisation de WDT doit se faire avec précaution de la réinitialisation (inattendue) répétée du programme. Pour éviter un « WDT time out » lors de l'exécution d'un programme, on a deux possibilités :

- Inhiber le WDT d'une façon permanente en mettant à 0 le bit WDTE dans l'EEPROM de configuration.
- Remettre le WDT à 0 périodiquement dans le programme à l'aide de l'instruction CLRWDT pour éviter son débordement.

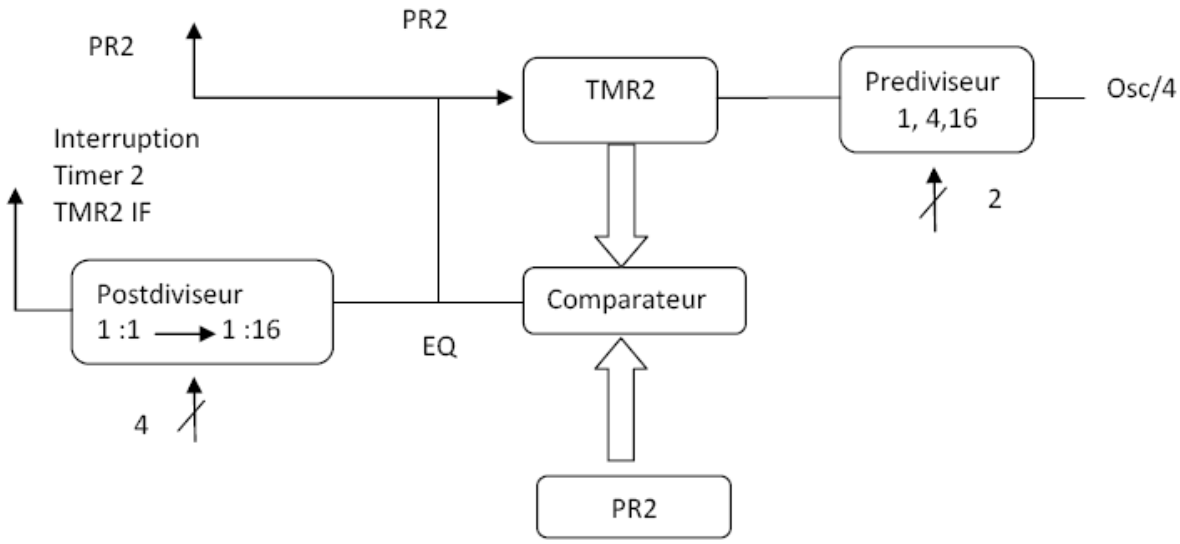
II.3.7 Les Mode CCP1 et CCP2 (Capture, Compare, PWM)

La PWM fait partie d'un bloc interne du PIC : le CCP ; ou Capture/Compare/PWM.
Les deux premiers modes assez flous et d'une approche assez difficile ne seront pas appliqué ici ; car très peu utilisés. Nous nous concentrerons sur la PWM.
La PWM (pulse width modulation) ou en français MLI (Modulation en Largeur d'Impulsion) est un signal auquel on fait varier la tension sans modifier ni l'amplitude ni la fréquence mais la largeur d'impulsion. Donc la PWM permet de gérer l'énergie transmise à l'extérieur. En effet, si un signal continu correspond à 100% d'énergie, un signal carré dont la durée d'état haut égale celle d'état bas correspond à 50% d'énergie. Le pourcentage d'énergie transmis se calcule en faisant le rapport de la durée d'état haut sur la durée de la période.
Ces signaux permettent en effet, de doser avec une grande efficacité la puissance appliquée à des charges alimentées sous une tension continue.
En utilisant la PWM, on peut facilement contrôler la vitesse de rotation d'un moteur à courant continu.



Le microcontrôleur PIC 16F877A est doté de deux modules CCP sur les broches 16 et 17 (CCP1 et CCP2). Pour utiliser le mode PWM, il convient de configurer les registres et les timers correspondant.

Le timer 2 permet de définir, grâce à son registre PR2 et à son pré-deviseur d'horloge, la fréquence de génération des signaux PWM qui sont prélèvé au niveau de sa sortie d'horloge. Cependant, pendant la phase de comptage, le registre de ce timer est comparé au contenu du registre de comparaison CCP1 et ; s'il y a égalité, la sortie PWM est remise prématurément a zéro alors que le comptage du timer 2 continue normalement. Ainsi si le registre PR2 est charge avec la valeur 100 alors que le registre CCP1 est charge avec la valeur 30, on produira des signaux PWM de rapport cyclique égale à 30%. Bien sûr, cela fonctionne avec n'importe quels autres contenus, le rapport cyclique étant simplement déterminé par les rapports des contenus de CCP1 et de PR2.



FigureII.5: synoptique fonctionnel de TIMER2

TMR2 : registre de comptage.

PR2 : registre de période.

II.4 les modes d'adressages

II.4.1 Adressage inhérent ou implicite

Le mnémonique de l'instruction mentionne la donnée sur laquelle porte l'opération (Contenu des registres), ou aucune donnée n'est nécessaire.

Exemple

CLRW : Mise a zéro de W.

NOP : Aucune opération (sert de temporisation).

SLEEP : Mise en sommeil du µC.

II.4.2 Adressage immédiat

L'instruction porte sur une valeur constante indiquée immédiatement après le mnémonique.

Exemple

MOVLW 0x80 : charge 0x80 dans W

ADDLW 0x14 : additionne 14 avec W et met le résultat dans W

II.4.3 Adressage direct

L'adressage de la mémoire de données se fait dans la Bank sélectionnée par les **BIT 5 (RP0)** et **BIT 6 (RP1)** du registre STATUS.

Exemple

MOVF VARIABLE, 0 ; Transfert le contenu de VARIABLE dans W.

MOVWF PORTB ; Transfert le contenu de W dans le registre Port B.

II.4.4 Adressage indirect ou indexé

Les Pics disposent a travers les registres **INDF** et **FSR** d'un mode d'adressage indexe, la structure est un peu particulière, **FSR** est le registre d'index et **INDF** permet d'accéder a son contenu.

Exemple :

MOVLW TAB_VAL ; Mettre dans W l'adresse de TAB_VAL

ADDLW 4 ; W devient W + 4

MOWF FSR ; Adresse w+ 4 dans le registre d'index FSR

MOVF INDF, 0 ; Transfert du contenu de TABLE [4] dans W

MOVWF PORTB ; Transfert du contenu de W sur le PORTB

II.5 Présentation du langage C de programmation des PICs

Evolué : le code est indépendant du processeur utilisé**.**

Typé : un type est l'ensemble des valeurs que peut prendre une variable.

Modulaire et structuré : tout programme est décomposable en tâches simples qui seront regroupées sous forme de groupe pour former des tâches plus complexes (structurés).

II.5.1 langage C du compilateur CCS

II.5.1.1 Les règles de bases

- Toutes instructions ou actions se terminent par un point virgule « **;** »

- Une ligne de commentaires doit commencer par **/*** et se terminer par ***/** ou commence par **//** norme C++.

- Un bloc d'instructions commence par **{**et se termine par**}.**

II.5.1.2 Les variables et les constantes

a. Les constantes : Nous déclarons une valeur constante en utilisant l'instruction **#define**

Syntaxe : < #define> <identificateur> <valeur> ; Exemple : #define PI 3,14;

.Déclarations spécifiques au compilateur CCS

- #bit id =x,y
  - Id : identifiant (Nom d'un bit)
  - X : Nom du variable ou d'une constante
  - Y : position

    Exemple : #bit RW =PORTA, 2

- #byte id = X
  - Id: identifiant
  - X: valeur 8 bits

Exemple : #byte PORTA = 5 // adresse du port A

b. Les variables

Les variables sont définies par « signé »ou « non signé »,

Syntaxe : <Signed><type><identificateur **1**>,.., <identificateur **n**>

.Les types du compilateur CCS

Type	**Taille**	**valeurs**
Int1	1 bit	0 OU 1
Int8	8 bits	De 0x00 à 0xFF ou 0 à 255
Int16	16 bits	De 0x0000 à 0xFFFF ou 0 à 65535
Int32	32 bits	De 0x0000000 à 0xFFFFFFFF
char	8 bits	De 0 à 255 ou une lettre A…….. Z
float	32 bits	Format flottant
short		Même type que int1
int		Même type que int8
long		Même type que int16

Exemples : Char MESSAGE; Int x,y,z ; …….. Non signés
Signed int A; Signed long NB; …… Signés

. Les bases du compilateur CCS

- Le décimal : A=10 ;
- L'octale : A=012 ;
- L'hexadécimal : A=0x0A ;
- Le binaire : A=0b00001010 ;

II.5.1.3 Les opérateurs du langage C

.L'opérateur d'affectation

Type	**Symbole**	**Exemple**
Opérateur d'affectation	=	X=10 ; Y = a + b

. Les opérateurs arithmétiques :

Type	Symbole	Exemple
Addition	+	a = a + b ; x = 5+ a
Soustraction	-	a = a – b ; y = c - 5
Moins unaire	-	a = - b
Multiplication	*	a = a * a ; b = y * 8
Division	/	c = 9 / b ; d = a / b
Reste de la division entière	%	r = a % b

. Les opérateurs de comparaison ou relationnels :

Type	Symbole	Exemple
Egalité	==	a == b ; c == 9
Différent	!=	c!= a
Supérieur	>	a > b ; 8 > a
Supérieur ou égal	>=	a >= b ; 8 >= a
Inférieur	<	a < b ; 8 < a
Inférieur ou égal	<=	a <= b ; 8 <= a

. Les opérateurs logiques :

Type	Symbole
Et logique	&&
Ou logique	||
Non logique	!

. Les opérateurs binaires bit à bit :

Type	Symbole	Exemple
Et binaire	&	x = y & z
Ou binaire	|	x = y | z
Ou exclusif	^	x = y ^ z

Complément à 1	~	x = b ~ z
Décalage de n bits à droite	>>	x = b >> n
Décalage de n bits à gauche	<<	x = b << n

II.5.1.4 Les structures répétitives.

Le langage "C" possède des instructions permettant de répéter plusieurs fois une même séquence en fonction de certaines conditions.

a. Structure "while" : tant que ... faire ...

Avec ce type d'instruction le nombre de répétitions n'est pas défini et dépend du résultat du test effectué sur la condition. Si cette dernière n'est jamais vérifiée, la séquence n'est pas exécutée.

```
while (int x !=0)
  {
  ...
  }
```

La structure précédente répète la suite d'instruction comprise entre crochets tant que la variable entière "x" est différent de 0.

b. Structure "do ... while" : faire ... tant que...

Cette structure ressemble fortement à la précédente à la seule différence que la séquence à répéter est au moins exécuter une fois même si la condition n'est jamais vérifiée.

```
do {
     ...
   }
 while (int x!=0);
```

c. Structure "for" : Pour <variable> allant de <valeur initiale> à <valeur finale> faire...

Cette instruction permet de répéter, un nombre de fois déterminé, une même séquence.

```
for (i=0;i<5;i++)
  {
  ...
  }
```

La structure précédente répète 5 fois la suite d'instruction comprise entre crochets. La variable "i" prendra les valeurs successives de : 0, 1, 2, 3 et 4.

d. Structure "if ... Else" : Si <condition> faire ... sinon faire ...

Avec cette structure on peut réaliser deux séquences différentes en fonction du résultat du test sur une condition.

```
if (a<b)   {c=b-a;}
else   {c=a-b;}
```

La structure précédente affecte la valeur "b-a" à "c" si "a" est inférieur à "b" sinon "c" est affecté par la valeur "a-b".

e. Structure "switch ... case".

Cette structure remplace une suite de "if ... else if ...else" et permet une de réaliser différentes séquences appropriées à la valeur de la variable testée.

```
switch (a)
   {
  case '1' : b=16;
  case '2' : b=8;
  case '3' : b=4;
  case '4' : b=2;
   }        // Dans cette structure "b=16" si "a=1", "b=8" si "a=4" etc.
```

II.5.1.5. La fonction d'interruption :

L'exécution d'une fonction d'interruption répond a un évènement qui peut être interne (périphérique : CAN, TIMER, EEPROM, USART, I2C) ou externe (RB0, PORTB) du microcontrôleur. L'appel d'une fonction d'interruption ne dépend pas de programme principal, mais elle l'interrompe pendant son exécution.

Une fonction d'interruption n'a pas de paramètre d'entrée et de sortie. Le compilateur CCS utilise une directive spéciale **INIT_XXXX** (**XXXX** nom de l'interruption) pour les différencier avec les autres fonctions logicielles.

Syntaxe : **#INIT_XXXX** //Nom de l'interruption

```
Void   nom de fonction (Void)
{
 Instruction 1 ;
 ……..
 Instruction n ;
  }
```

II.5.1.6. Quelques directives et fonctions du Pic C Compiler

1- <u>Directives</u>

#use delay

Syntaxe : #use delay (***clock=fréquence***)

Rôle : Renseigne le compilateur sur la fréquence du quartz utilisé.

Exemple : #use delay(clock=4000000)

#fuses

Syntaxe : #fuses ***options***

Rôle : Permet de définir le mot de configuration. Les options sont :

- LP, XT, HS, RC
- WDT, NOWDT
- PUT, NOPUT
- PROTECT, NOPROTECT.

Exemple : #fuses XT, NOWDT, NOPUT, NOPROTECT

#int_xxxx

Syntaxe : #int_ext : interruption externe.

#int_ RB : changement d'état de RB4 à RB7.

#int_TIMER0 : débordement du timer0.

#int_EEPROM : fin d'écriture dans l'EEPROM.

Rôle : Spécifie la source de l'interruption.

2- Fonctions

BIT_CLEAR ()

Syntaxe : BIT_CLEAR (***var, bit***)

Rôle : Mettre à 0 le bit « bit » de la variable « var ».

Exemple : a=0x1F

BIT_CLEAR (a, 3), donc a devient 17 hexa.

BIT_SET ()

Syntaxe : BIT_SET (***var, bit***)

Rôle : Mettre à 1 le bit « bit » de la variable « var ».

Exemple : a=0x1F

BIT_SET (a, 6) , donc a devient 3F hexa.

BIT_TEST ()

Syntaxe : BIT_TEST (***var, bit***)

Rôle : Teste l'état du bit « bit » du variable « var ».

Exemple : a=0x1F

BIT_TEST (a, 2)

Le résultat est 1 car le bit 2 de la variable « a » est à 1.

DELAY_MS ()

Syntaxe : DELAY_MS(***x***)

Rôle : Temporisation se « x » ms.

Exemple : DELAY_MS(2000)

Temporisation de 2 secondes.

INPUT ()

Syntaxe : etat=INPUT (***pin***)

Rôle : Permet de lire l'état d'une broche d'un port préalablement configurée en entrée.

Exemple : a=input(PIN_B0)

INPUT_x ()

Syntaxe : etat=INPUT_x ()

Rôle : Permet de lire l'état d'un port (x sur 8 bits) préalablement configuré en entrée.

Exemple : c=input_A()

OUTPUT_x ()

Syntaxe : OUTPUT_x (***valeur***)

Rôle : Permet de sortir l'octet «valeur» sur le port x, préalablement configuré en sortie.

Exemple : output_B(0x1F)

OUTPUT ()

Syntaxe : OUTPUT (***pin***, ***etat***)

Rôle : Permet de mettre la pin (***pin***) à l'état logique (***etat***).

Exemple : Output (PIN_A3,1)

OUTPUT_HIGH ()

Syntaxe : OUTPUT_HIGH (***pin***)

Rôle : Permet de mettre à 1 la pin (***pin***).

Exemple : output_high(PIN_A3)

OUTPUT_LOW ()

Syntaxe : OUTPUT_LOW (***pin***)

Rôle : Permet de mettre à 0 la pin (***pin***).

Exemple : output_low (PIN_A3)

ROTATE_LEFT ()

Syntaxe : ROTATE_LEFT (***adresse***, ***n***)

Rôle : Rotation à gauche de « n » positions de l'octet ayant pour adresse « adresse ».

Exemple : a=0x86

ROTATE_LEFT (a, 1), donc a devient 0d hexa.

ROTATE_RIGHT ()

Syntaxe : ROTATE_RIGHT (***adresse***, ***n***)

Rôle : Rotation à droite de « n » positions de l'octet ayant pour adresse « adresse ».

Exemple : a=0x86

ROTATE_ RIGHT (a, 1), donc a devient 43 hexa.

SET_TRIS_x ()

Syntaxe : SET_TRIS_x (***valeur***)

Rôle : Configure la direction du port « x ».

Exemple : SET_TRIS_B (0x0F)

Les pins B7, B6, B5, B4 sont configurées en sortie (0).

Les pins B3, B2, B1, B0 sont configurées en entrée (1).

SHIFT_LEFT ()

Syntaxe : SHIFT_LEFT(***adresse, n, bit***)

Rôle : Décalage à gauche de « n » positions de l'octet d'adresse « adresse » ; bit est le bit introduit 0 ou 1.

Exemple : SHIFT_LEFT (b, 2, 0)

Décalage à gauche de 2 positions de la variable « b ».

SHIFT_RIGHT ()

Syntaxe : SHIFT_RIGHT(***adresse, n, bit***)

Rôle : Décalage à droite de « n » positions de l'octet d'adresse « adresse » ; bit est le bit introduit 0 ou 1.

Exemple : SHIFT_ RIGHT (b, 2, 0)

Décalage à droite de 2 positions de la variable « b ».

II.6 Conclusion

Partant d'une présentation générale sur les microcontrôleurs, nous avons ensuite défini la famille des PICs et plus particulièrement le 16F877A.

En conclusion dans ce chapitre nous pouvons dire que le microcontrôleur 16F877A peut bien jouer le rôle d'une unité de contrôle pour notre système.

Maintenant, nous pouvons passer à l'étude et la conception puisque le composant le plus important dans notre carte est déjà familier.

CHAPITRE III

LE ROBOT RÉALISÉ

Unité De Perception

III.1 Introduction

Avant de s'intéresser plus précisément aux ultrasons, voyons quelles sont les autres technologies disponibles. Tous les capteurs permettent de se représenter l'environnement (contact, odométrie, sensibilité à la lumière, au son, à la chaleur…). Certains capteurs permettent de mesurer à distance les obstacles et ainsi permettre au robot de prendre une décision en fonction des obstacles (ou l'absence d'obstacle autour de lui) ; On parle alors de télémètre, c'est-à-dire la mesure à distance d'un point éloigné. En outre les ultrasons, deux autres technologies de télémétrie permettent d'effectuer ces mesures: L'Infrarouge et Le Laser.

Les capteurs à ultrasons, appelés également télémètres à ultrasons, font partie des capteurs qui permettent de mesurer les distances sans contact. Ils sont relativement bien présents dans la communauté robotique par leur bon rapport efficacité/prix.

Le télémètre à ultrasons est basé sur la mesure du temps écoulé entre l'émission et le retour de l'écho. Lorsque l'onde ultrasonore est émise elle se propage à la vitesse du son, dans l'air environ à 340 m/sec. Dès qu'un obstacle est rencontré, l'écho revient vers le transducteur ce qui permet de calculer alors le temps écoulé entre l'émission et la réception de l'onde. Ces ondes sont inaudibles par l'oreille humaine.

III.2 La notion de perception

La notion de perception en robotique mobile est relative à la capacité du système à recueillir, traiter et mettre en forme des informations utiles au robot pour agir et réagir dans le monde qui l'entour. Alors que pour des tâches de manipulation on peut considérer que l'environnement du robot est relativement structuré, ce n'est plus le cas lorsqu'il s'agit de naviguer de manière autonome dans des lieux très partiellement connus. Aussi, pour extraire les informations utiles à l'accomplissement de sa tâche, il est nécessaire que le robot dispose des capteurs mesurant aussi bien son état interne que l'environnement dans lequel il évolue. Le choix des capteurs dépend bien évidemment de l'application envisagée.

III.3 Caractéristiques des différentes technologies de télémétrie

Le tableau suivant illustre les principaux avantages et inconvénients des différentes solutions proposées par les constructeurs de robots mobiles

	Ultrason	Infrarouge	Laser
Portée	De 1 à 350 cm	De 5 à 80 cm	Plusieurs mètres à plusieurs dizaines de mètres selon les modèles.
Directivité	Cône d'environ 30° à 40°	Cône d'environ 5°	Les plus directifs (de l'ordre du degré, voire du demi-degré)
Précision	Relativement précis mais la précision diminue avec la distance, l'angle de mesure et les conditions de température et de pression.	Relativement précis mais la précision diminue avec la distance.	Sont précis avec un bruit de quelques centimètres sur des mesures de plusieurs mètres.
Coût	Peu chers	Peu chers	Relativement chers
Sensibilité aux interférences	Sensible à la température et à la pression. Egalement sensible aux autres robots utilisant la même fréquence ce qui peut poser problème dans une compétition.	Sont sensibles aux fortes sources de lumière qui contiennent un fort rayonnement infrarouge. Sont également sensibles à la couleur et à la nature des obstacles.	Ne peut pas détecter les objets réfléchissant la lumière laser (vitres, objets chromés,…)

III.4 Son et ultrasons

Le son est une onde mécanique et élastique se propageant dans un milieu physique sous forme d'ondes longitudinales ou de compression. Ce phénomène est par exemple mis à profit par les hauts parleurs qui font vibrer une membrane qui à son tour fait vibrer l'air. Le son se propage d'autant plus vite que le milieu est dense, ce qui explique que le son soit plus rapide dans l'eau que dans l'air. Ceci explique également que les capteurs à ultrasons ne fonctionnent pas dans le vide car le son ne s'y propage pas. Vous ne verrez donc pas de capteurs à ultrasons sur les sondes spatiales.

Les ultrasons ont une fréquence supérieure à 20 k Hz et sont donc inaudibles par l'homme. On distingue deux types d'ultrasons selon la gamme de fréquence :

- Les ultrasons de faible puissance qui sont utilisés pour la mesure de distance (télémétrie), le contrôle non destructif, l'échographie et l'acoustique sous-marine. C'est ce type d'ultrasons qui nous intéresse ici
- Les ultrasons de forte puissance qui modifient selon leur action, le milieu dans lequel ils se propagent. Ces actions peuvent être mécaniques, thermique ou chimique.

III.5 Les ultrasons dans la nature

De nombreux animaux peuvent entendre les ultrasons comme le chien ou la chauve-souris. La chauve-souris a aussi la particularité de pouvoir en émettre dans le but de se repérer, on parle d'écholocalisation. C'est exactement ce principe qui est exploité par les roboticiens pour planifient les trajectoires des robots mobiles. Ces animaux utilisent les ultrasons non seulement pour se repérer mais aussi pour localiser leurs proies et aussi communiquer.

L'homme a mis à profit l'écholocalisation dans diverses activités. L'une des plus connue est l'échographie qui permet de voir certains tissus vivants en fonctionnement (par exemple le fœtus dans le ventre de la mère ou les ligaments des articulations). L'autre utilisation très connue est le sonar utilisé par les navires militaires et les sous-marins.

III.6 Fonctionnement des capteurs à ultrasons

L'émetteur émet un son à une fréquence définie (généralement autour de 40 kHz) et le récepteur collecte le son répercuté par les obstacles. La distance aux objets est calculée par le temps mis par le son pour revenir au récepteur.

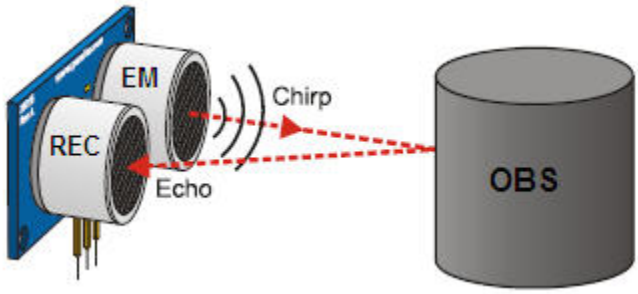


Figure III.1: fonctionnement de capteur à ultrason

La forme du faisceau est la caractéristique du capteur utilisé. La Figure suivante présente une forme typique de faisceau d'ultrasons.

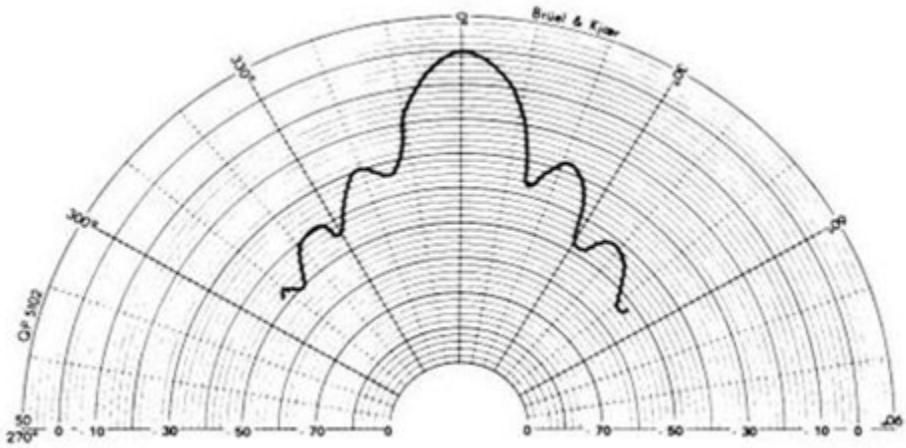


Figure III.2: forme typique d'un faisceau d'ultrason

On remarque que l'angle effectif de fonctionnement est d'environ 30° (ce qui est important comparé aux autres types de capteurs) avec des lobes secondaires moins importants de part et d'autre. La mesure sera ainsi plus précise dans le cône central de 30° et sera moins précise sur les parties latérales. Ceci explique que généralement les capteurs à ultrasons sont montés sur les parties rotatives afin que différentes mesures puissent être effectuées en utilisant la partie centrale du cône de visualisation.

III.7 Les différentes zones de détection d'un capteur à ultrason

III.7.1 Zone aveugle

Il faut tenir compte du fait qu'à très courte distance, les capteurs ultrasons sont aveugles. Ceci est dû à la temporisation entre l'émission de l'onde sonore et de début de la détection de l'onde réfléchie qui est nécessaire pour ne pas perturber cette mesure. Cette zone comprise entre la face sensible du détecteur et la portée minimale, donc il faut éviter tout passage d'objets dans cette zone pendant le fonctionnement normal du capteur, cela peut provoquer un état instable de sortie.

III.7.2 Zone de détection

Zone dans laquelle le capteur est sensible. Selon les modèles de capteur cette zone peut être ajustable ou fixe.

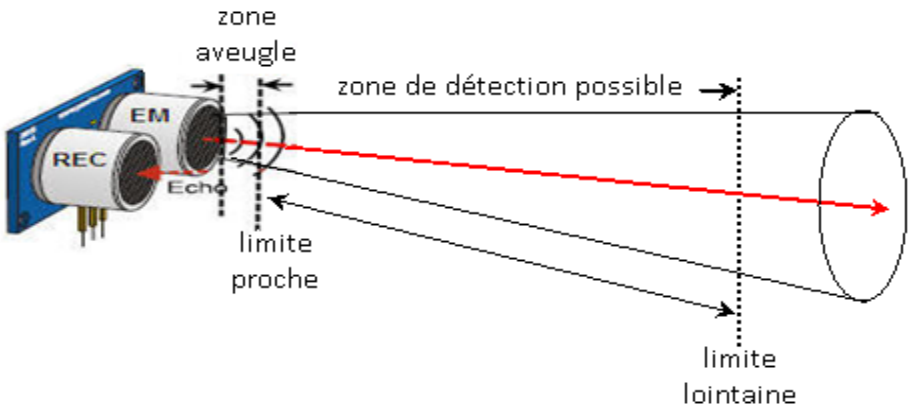
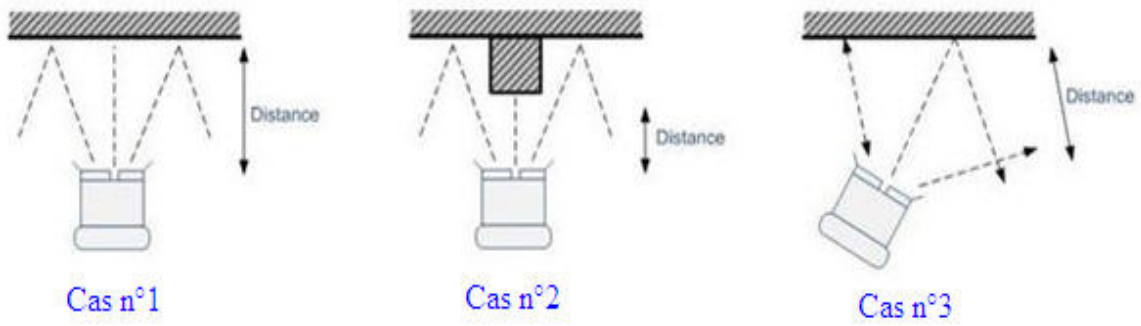


Figure III.3: Les différentes zones de détection d'un capteur à ultrason

La figure ci-dessous présente trois cas de mesure classique.



1ier cas : Le premier cas générera une mesure précise car le capteur est bien en face et perpendiculaire à l'obstacle.

2ième cas : générera également une mesure précise mais donnera une « vue » de l'obstacle situé directement en face du capteur.

3ième cas : générera une mesure imprécise étant donné que c'est la partie latérale gauche du capteur qui procède à la mesure.

III.8 Quelques limitations

La forme des obstacles joue un rôle essentiel, car elle peut amener le robot à ne pas se représenter correctement son environnement.

III.8.1 La forme des obstacles

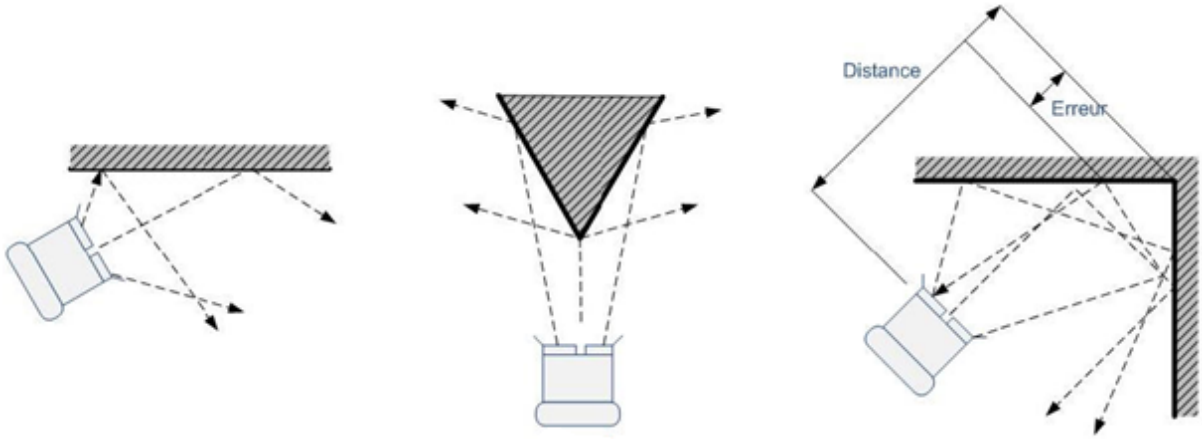


Figure III.4: quelques limitations sur les capteurs à ultrasons

Comme la montre la figure ci-dessus, il faut tenir compte des erreurs générées par la forme des obstacles.

III.8.2 La texture de l'obstacle

Elle joue également un rôle important. Un mur recouvert de moquette réfléchira moins bien l'onde qu'un mur recouvert de peinture uniquement.

III.8.3 Le cross-talk

Deux capteurs à ultrasons ne peuvent pas être utilisés côte à côte car dans ce cas, il n'est pas possible, s'ils ont la même fréquence, de distinguer lequel des deux a émis une onde. On parle de phénomène de cross-talk. Une solution pour un robot qui possède plusieurs capteurs à ultrasons, c'est d'activer les capteurs les uns après les autres, ce qui diminue le taux de rafraîchissement global.

III.9 Caractéristiques des ondes ultrasoniques :

- ✓ Fréquence supérieur à 20khz, généralement, elles sont véhiculées par le milieu ambiant, qui est l'air, à la vitesse de 340m/s.
- ✓ La figure suivante représente les différentes gammes du son :

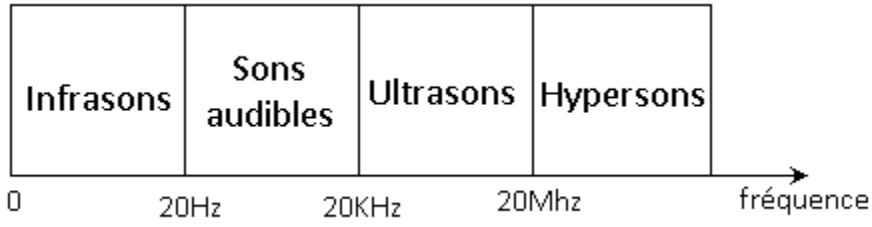


Figure III.5 : les différentes gammes du son.

- ✓ Le tableau suivant représente la vitesse de propagation (la célérité) des ondes ultrasoniques dans les différents milieux, en m/s:

Milieu	Vitesse (m/s)
Air	340
Eau	1480
Sang	1570
Aluminium	6400

Figure III.6 : la célérité des ultrasons dans les différents milieux

III.10 Télémétrie directe par mesure de temps de vol

On utilise cette méthode pour localiser notre robot par rapport à des obstacles statiques ou dynamiques, cette méthode repose sur l'idée de calculer le temps de vol d'un signal ultrasonique depuis son point de départ (émetteur) jusqu'à l'obstacle puis le chemin de retour vers le point de départ (récepteur) Comme le montre la figure suivante :

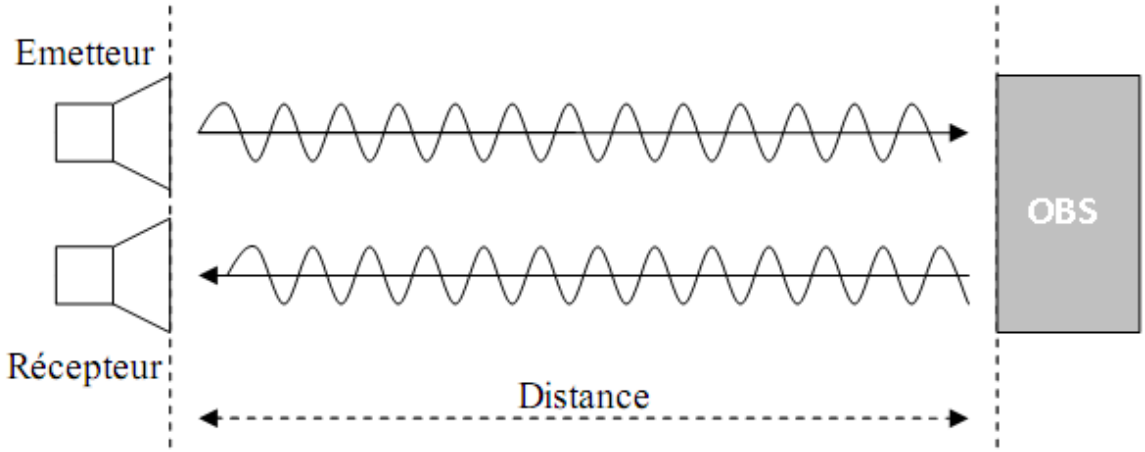


Figure III.7 : Schéma synoptique d'un télémètre à ultrason

Alors si la distance entre le transducteur et l'obstacle est '**D**', donc la distance parcourue par l'onde ultrasonique est '**2D**'

Le temps du parcoure est donné par : $\mathbf{T = \frac{2D}{Vson}}$

Où **T** : le temps entre l'émission et la réception.

D : la distance entre le transducteur et l'obstacle.

Vson : la vitesse de déplacement des ultrasons dans l'air.

Généralement, l'air est la porteuse de l'onde ultrasonique, cette dernière se propage, alors, à la même vitesse que le son (célérité de l'onde sonore). Physiquement et mathématiquement la vitesse du son ne dépond pas de la fréquence du signal sonore, mais du type du milieu, de la pression et de la température suivant la relation mathématique suivante :

$$\mathbf{V}\text{son} = \sqrt{\frac{\overline{E}}{\rho}} = \sqrt{\frac{\overline{Y.P}}{\rho}}$$

Tel que : **V**son : célérité de son.

E =Y.P : coefficient d'élasticité du gaz.

ρ : Densité de l'air (1.293 kg/m^3).

III.11 Caractéristiques du capteur à ultrason Ping))) utilisé

Avec le capteur à ultrason PING))), on peut calculer la distance sans contact avec les obstacles, la portée de ce capteur est comprit entre 2 cm à 3 mètres. Son utilisation est très facile, si on le relie à un microcontrôleur, car il demande un seul I/O pin.

Le capteur à ultrason Ping))) fonctionne de la même façon qu'un sonar ; C'est-à dire qu'une pulsation est envoyée et le capteur attend l'écho du signal. En déterminant le délai de l'écho, on peut déterminer la distance d'un objet.

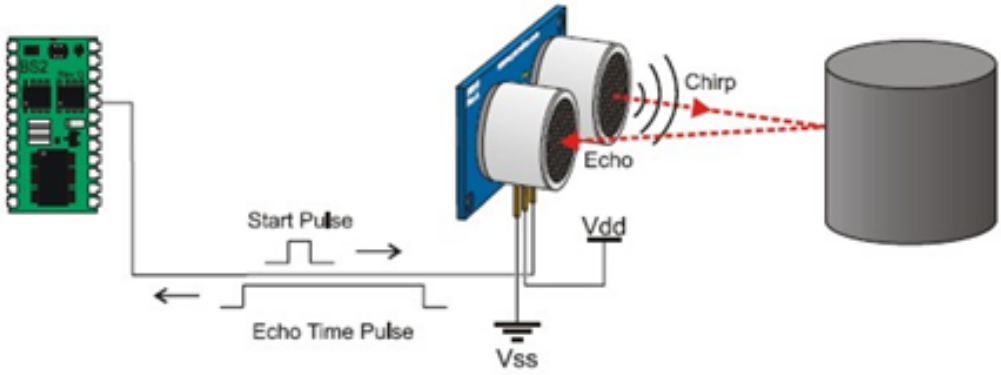


Figure III.8 : Schéma de fonctionnement du capteur PING))) avec un PIC.

III.11 .1 Caractéristiques techniques

- ✓ Tension d'alimentation : 5VDC.
- ✓ Courant d'alimentation : 30mA typique, 35mA max.
- ✓ La portée : 2cm à 3m.
- ✓ L'impulsion d'entrée : impulsion positive TTL, 2µs min, 5µs typique.
- ✓ L'impulsion de l'écho : impulsion positive TTL, 115µs à 18.5ms.
- ✓ La fréquence des ondes sonores : 40KHz
- ✓ Il a une led indicatrice de fonctionnement de capteur.
- ✓ Temporisation entre les mesures : 200µs.
- ✓ Poids : 9 grammes.

III.11 .2 Branchement du capteur à ultrason Ping)))

Le module de capteur à ultrason Ping))) contient 3 pins (broches) :

- ✓ GND : la masse (Vss).
- ✓ 5V : 5VDC (Vdd).
- ✓ SIG : signal (I/O pin).

Branchez le capteur selon les images ci-dessous :

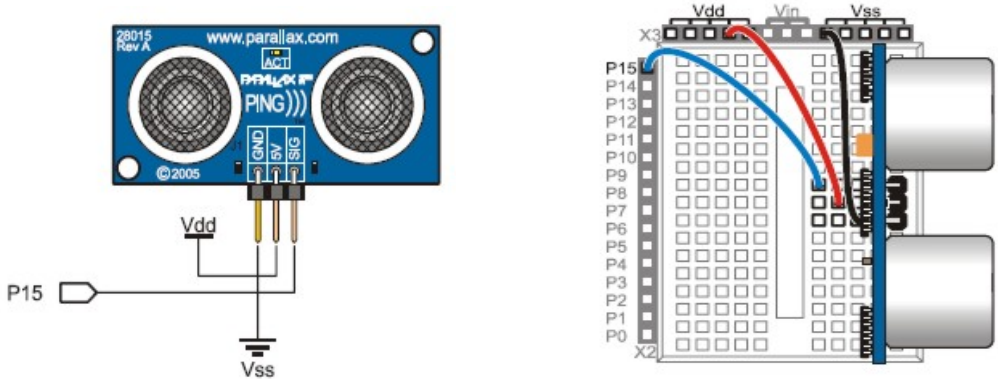


Figure III.9 : branchement du capteur à ultrason PING))).

III.11 .3 limitation du capteur à ultrason Ping)))

Le capteur à ultrason PING))) a certaine limitation physique :

a) La distance maximale est environ 3.3m.

b) L'angle formé par la surface de l'objet et le capteur doit être supérieur à 45 degré.

c)I ne peut pas capter les objets trop petits.

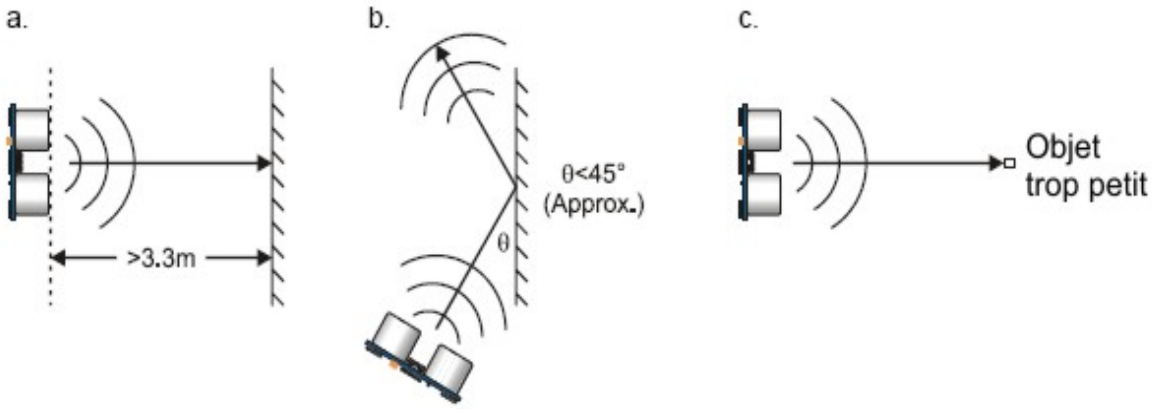


Figure III.10 : limitations physiques du capteur à ultrason PING))).

III.11 .4 Le protocole de communication avec le capteur à ultrason Ping)))

Pour que le capteur à ultrason PING)) fonctionne, on injecte une impulsion (trigger pulse) de 2µs à 5µs au capteur par le microcontrôleur, ensuite, le capteur émet une onde sonore de 40KHz (ultrasonic burst), cette onde traverse l'air à une vitesse environ 340m/s, dès qu'un obstacle est rencontré, l'écho revient vers le transducteur qui calcul alors le temps écoulé entre l'émission et la réception de l'onde.

Le capteur PING))) va fournir une impulsion en sortie, vers le microcontrôleur, dont sa largeur est le temps de vol de signal ultrasonique depuis son point de départ (émetteur) jusqu'à l'obstacle puis le chemin de retour vers le point de départ (récepteur).

Avec ce temps de vol, on peut facilement calculer la distance de l'obstacle en utilisant la relation suivante :

$$\mathbf{T}=\frac{\mathbf{2D}}{\mathbf{V}\text{son}} \quad \text{d'où :} \quad \mathbf{D}=\frac{\mathbf{T}}{\mathbf{2}} * \mathbf{V}\text{son}$$

Où **T** : temps de vol.

D : la distance entre le transducteur et l'obstacle.

Vson : la vitesse de propagation des ultrasons dans l'air.

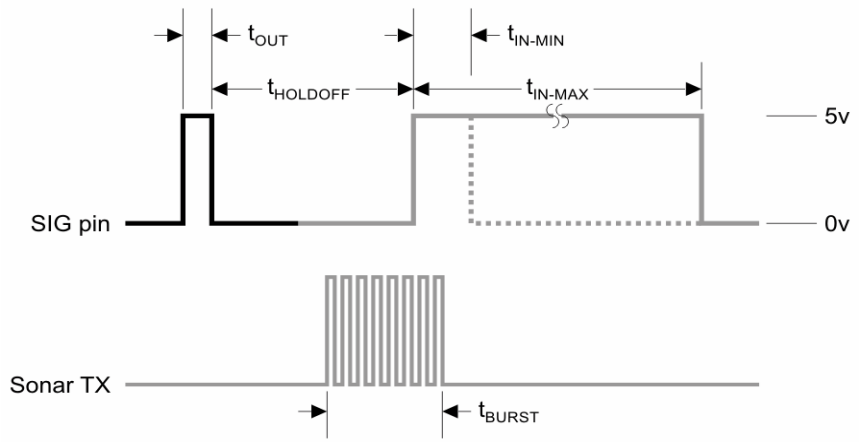


Figure III.11 : protocole de communication avec le capteur PING))).

▬	microcontrôleur	Impulsion d'entrée (Trigger pulse)	tOUT	2 µs (min), 5 µs typique
	Le capteur PING)))	Echo Hold-off	tHOLD	750 µs
		Fréquence de l'onde (BURST)	tBURST	200 µs @ 40 kHz
		Impulsion Echo minimale	tIN-	115 µs
		Impulsion Echo maximale	tIN-	18.5 ms
		Temporisation avant la mesure suivante		200 µs

III.11 .5 La forme de faisceau du capteur à ultrason Ping))) (le cône de détection)

On voie que l'angle effectif de fonctionnement est de 40°. La mesure sera ainsi plus précise dans le cône central de 40° et sera moins précise sur les parties latérales. Ceci explique que généralement les capteurs à ultrasons sont montés sur les parties rotatives afin que différentes mesures puissent être effectuées en utilisant la partie centrale du cône de visualisation.

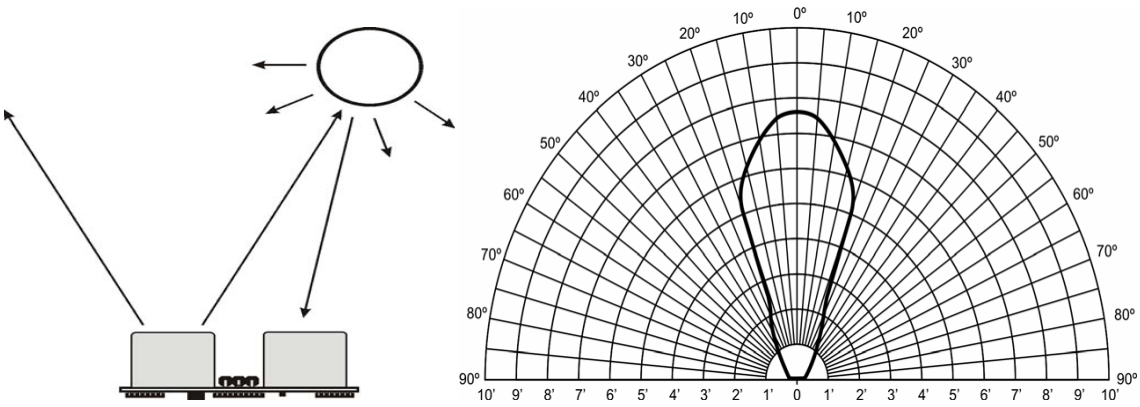


Figure III.12: le cône de détection du capteur à ultrason Ping)))

III.11 .6.1 Comment calculer le temps de vol (temps de l'écho) ?

Pour cela dans notre programme, nous avons utilisé le compteur TIMER1 qui s'incrémente automatiquement à la fréquence de travail de microcontrôleur PIC, qui est 1/4 de la fréquence d'horloge ; dans notre cas la fréquence d'horloge utilisé est de : 4MHz, donc le compteur TIMER1 s'incrémente chaque 1MHz ce qui donne chaque 1µs. En utilisant l'interruption de TIMER1 pour calculer les débordements, et autres instructions de compilateur PIC C Compiler, on peut facilement calculer le temps de vol.

III.11 .6.2 Comment avoir la distance de l'obstacle ?

Une fois, nous avons le temps de vol, on peut facilement calculer la distance de l'obstacle en utilisant la relation mathématique précédente :

$$\mathbf{D} = \frac{\mathbf{T}}{\mathbf{2}} * \mathbf{V}\text{son}$$ Où **T** : temps de vol.

D : la distance entre le transducteur et l'obstacle.

Vson: 340m/s = 0.034cm/µs.

III.11 .7 Représentation de l'impulsion émet par le transducteur et l'écho reçu

La figure suivante présente le protocole de communication du capteur PING))) avec le microcontrôleur PIC, ainsi le signal émet par ce capteur dans l'air (ultrasonic burst), et la forme de signal de l'écho reçu.

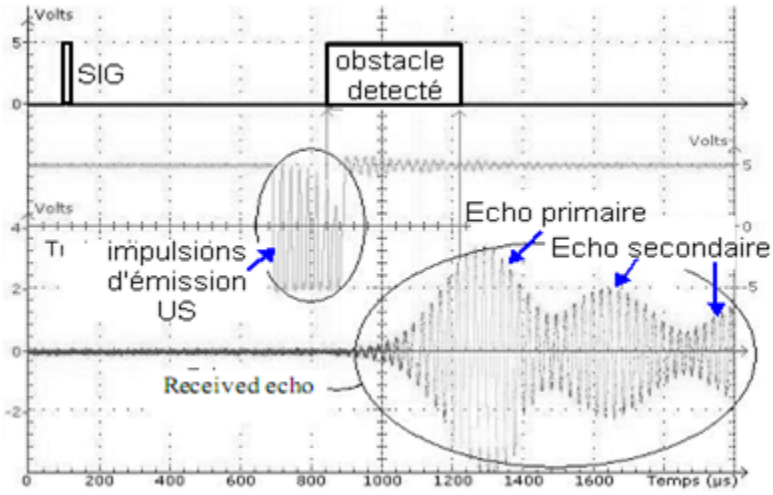


Figure III.13: la réponse du capteur à ultrason PING)))

III.11 .8 Conclusion

Tous les télémètres à ultrasons fonctionnent de la même façon, quelque soit le constructeur. Les différences vont se situer dans les fréquences émises, et le cône de diffraction de l'onde. Selon les modèles, les capteurs à ultrasons seront alors plus ou moins précis, et auront une portée plus ou moins grande. Généralement, plus le cône est fin, plus le capteur sera précis, plus il sera cher.

Les capteurs à ultrasons sont faciles à manipuler et permettent, combinés à d'autres capteurs, d'apporter des informations très utiles sur l'environnement du robot. A n'en pas douter, vos robots aurons un ou plusieurs capteurs de ce type !

CHAPITRE IV

LE ROBOT RÉALISÉ

Télécommande De Pilotage

IV.1 Introduction

Ce chapitre, décrit la réalisation d'une télécommande infrarouge, en vue de commander à distance notre chariot mobile à quatre roues. Cette télécommande est composée essentiellement de deux modules (Emetteur – récepteur). L'émetteur est conçu à base du PIC 16f84A et d'un clavier de six touches, dont le rôle est d'émettre le code approprié à une touche appuyé, qui est modulés sous le protocole RC5 afin de former une trame qui sera envoyée via une diode infrarouge. Quant au récepteur qui est conçu à base du PIC 16f877A, a pour rôle la réception et le décodage du faisceau infrarouge, ensuite donne l'ordre d'exécuter la tâche associée à la touche appuyée.

IV.2 Conception matérielle de la télécommande infrarouge

IV.2.1 Le principe général de fonctionnement de la télécommande

La carte de la télécommande qu'on a réalisée, peut être décomposée en plusieurs parties, comme le montre les schémas suivants :

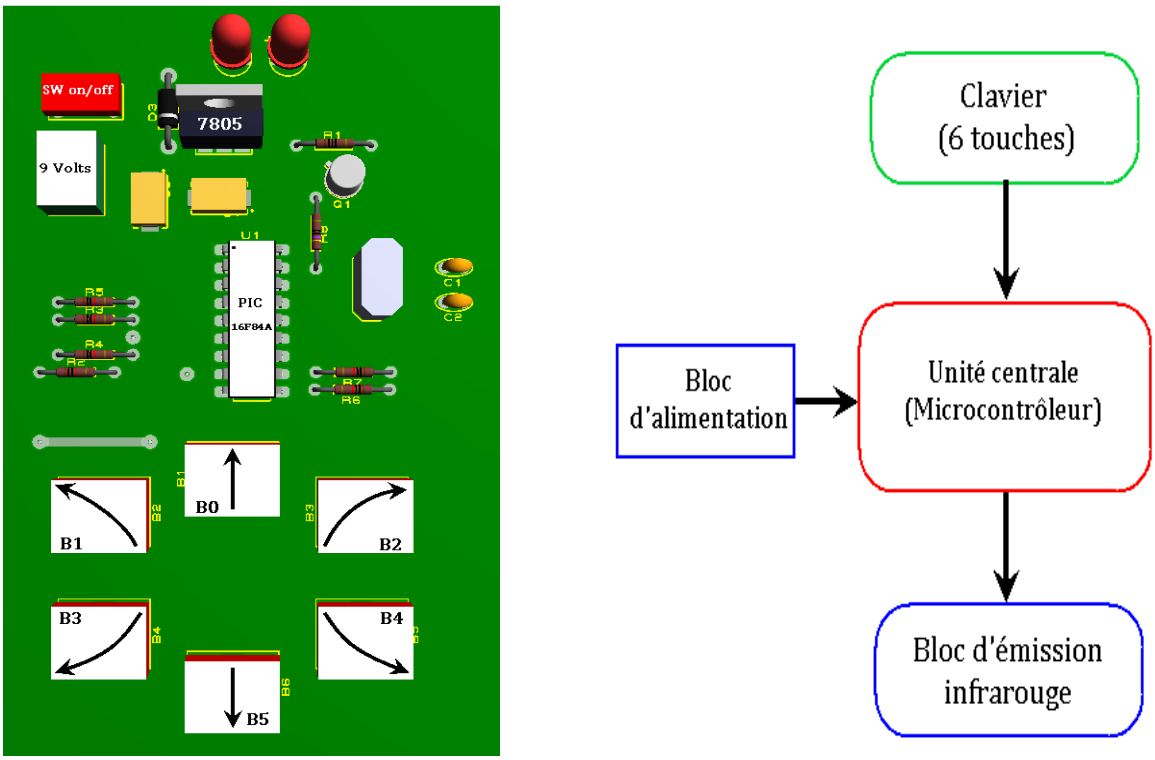


Figure IV.1 : (a) Schéma en 3D de la télécommande (b) Schéma synoptique

A partir de ces schémas, on distingue les différents blocs de notre télécommande :

- Bloc d'alimentation.
- Le microcontrôleur PIC16F84A.
- Le clavier de 6 touches.
- Bloc d'émission infrarouge.

IV.2.2 Bloc d'alimentation

L'alimentation logique +5 volts représentée sur la figure IV.2, est obtenue à partir de l'alimentation 9 volts réglée par le régulateur de tension LM7805 qui délivre une tension de 5 volts pour une tension d'entrée 36V>Vin>7.5V.

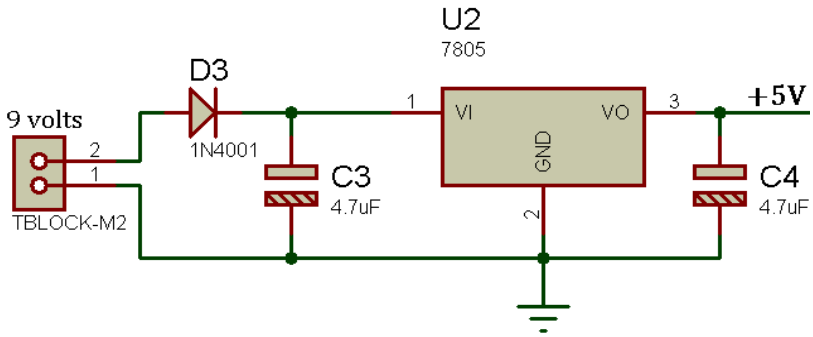


Figure IV.2 : Circuit d'alimentation de la télécommande

IV.2.3 Le microcontrôleur PIC16F84A:

Performance du PIC 16F84A :

- ✓ Un jeu de 35 instructions.
- ✓ Fréquence d'horloge pouvant aller jusqu'à 20MHz.
- ✓ Une mémoire programme de 1024mots de 8bits.
- ✓ Une RAM de données de 68 octets.
- ✓ Une EEPROM de 64 octets.
- ✓ Une pile à 8 niveaux.
- ✓ Trois modes d'adressages: immédiat, direct, indirect.
- ✓ 4 sources d'interruptions :

  Front sur RB0/INT, Débordement du TIMER0,

  Fin d'écriture en EEPROM, Changement de front sur RB4 :RB7.

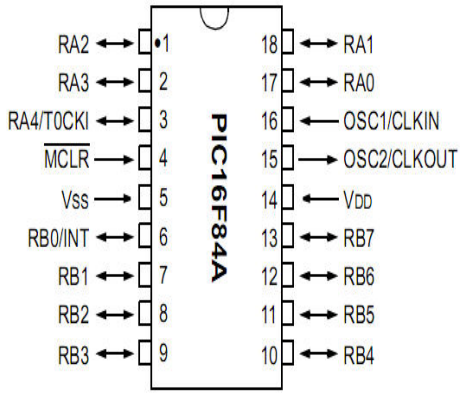
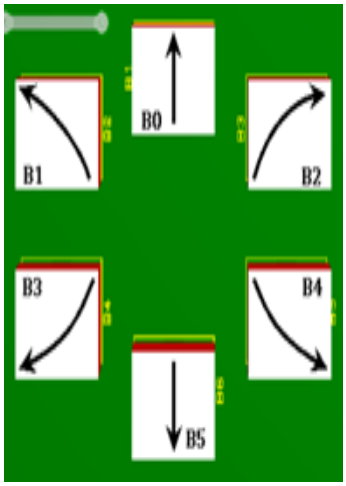


Figure IV.3 : Brochage du PIC 16F84A

IV.2.4 Le clavier:

Le clavier contient 6 touches, qui nous permettent de choisir le code à envoyer, sachant que chaque code est associé à une manœuvre au niveau du robot comme indiqué par le tableau suivant :



la touche	La manœuvre associé
B0	marche avant
B1	tourne à gauche en avant
B2	tourne à droite en avant
B3	tourne à gauche en arrière
B4	tourne à droite en arrière
B5	marche arrière

Figure IV.4 : le clavier

IV.2.5 Bloc d'émission infrarouge

Ce circuit contient un transistor de type 2N2222 et deux diodes infrarouges montées en série, pour avoir une longue portée du signal à envoyer.
Ces diodes infrarouges sont reliées au microcontrôleur via le transistor comme le montre la figure suivante, dans le but d' émettre un signal de 38 KHz.

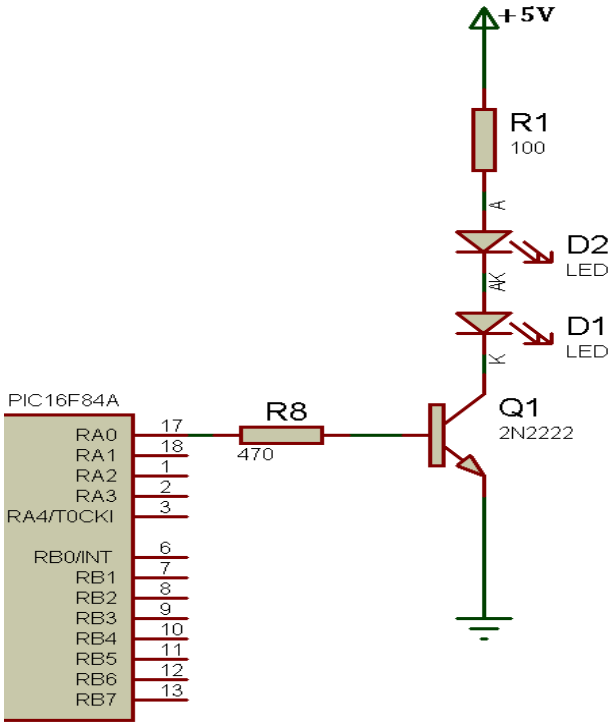


Figure IV.5 : Circuit d'émission infrarouge

IV.2.6 Le transistor 2N2222

Le rôle de ce transistor dans cette application est l'amplification du signal provenu du microcontrôleur vers les diodes infrarouges. Cette amplification est imposée, car le pic ne fournit que 30mA au maximum qui est insuffisant pour le bon fonctionnement. Son schéma de principe est donne par la figure suivante :

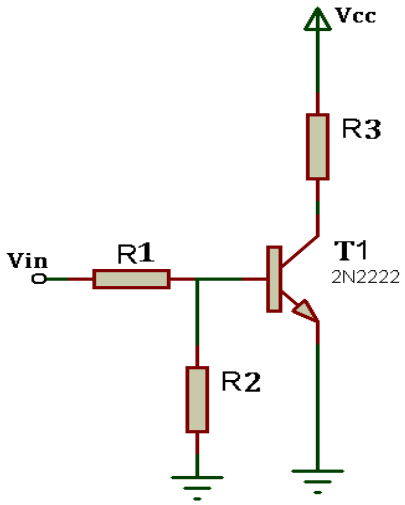


Figure IV.6 : schéma de principe de transistor 2N2222

IV.2.6.1 Calcul de la résistance de base :

La résistance R1 limite le courant de base du transistor. Pour ne pas claquer le transistor, **Ib** doit être de l'ordre du mA. Le calcul de cette résistance ne fait appel qu'à la loi des mailles :

Vin=R1.Ib+Vbe.

Vbe est la tension inverse d'une diode soit 0.7 V ; On déduit donc :

R1= (Vin-Vbe)/Ib.

R1 = (5-0.7)/ 10^{-2}= 430 Ohm (cette résistance est calculée pour un Ib=10mA).

Dans notre cas, on a pris une résistance proche de R=470 Ohm.

IV.2.7 la carte électronique de la télécommande :

Le schéma électrique de l'émetteur est donné par cette figure :

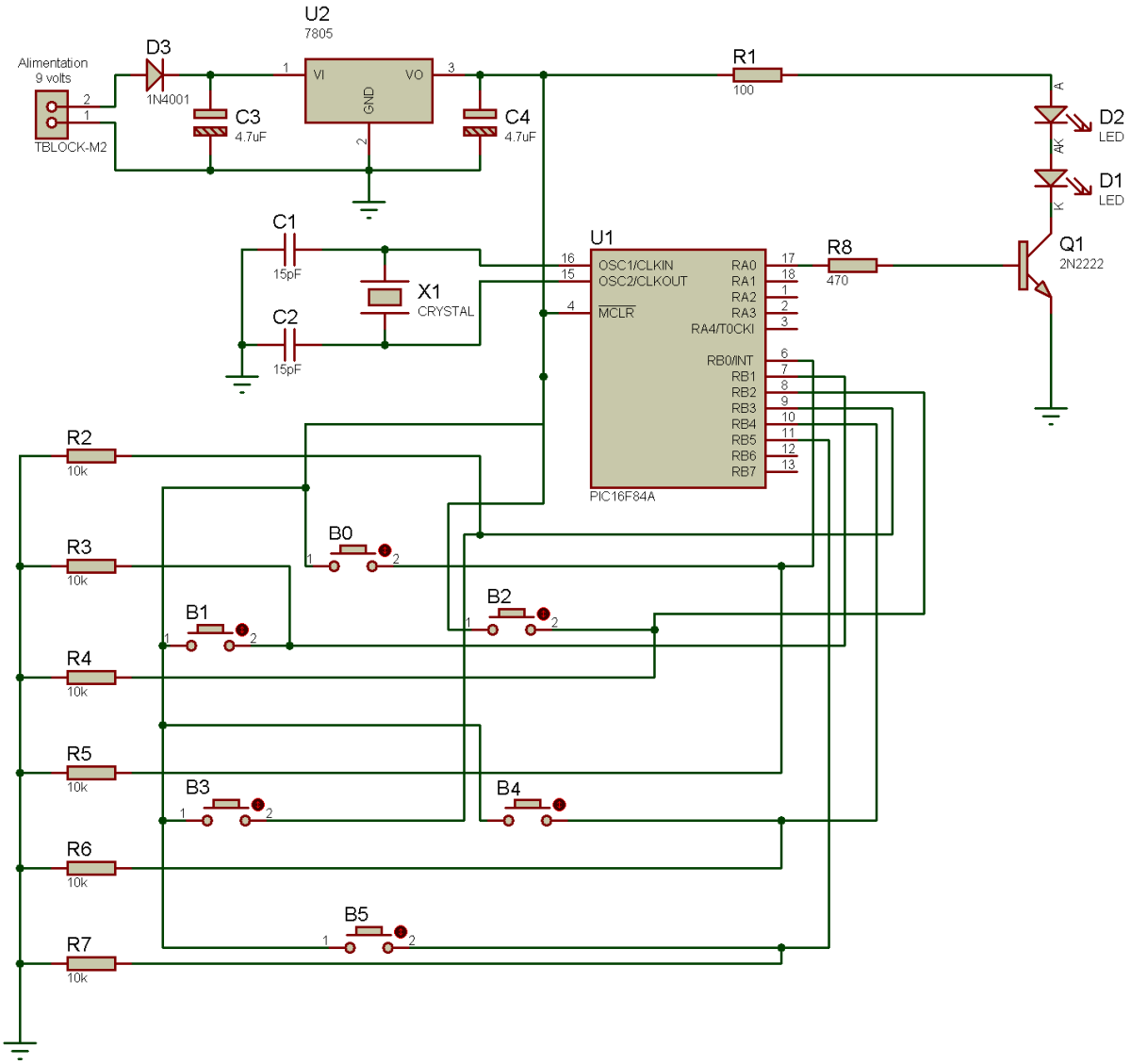


Figure IV.7 : Schéma électrique de l'émetteur

IV.3 Conception logicielle de la télécommande infrarouge

Après avoir présenté en détails les différents blocs constituant la télécommande infrarouge du robot ; Maintenant, on passe à la deuxième partie, où on va représenter protocole de communication RC5, et les différents organigrammes de fonctionnement de la télécommande.

IV.2.1 L'infrarouge et le protocole de transmission RC5

IV.2.1.1 Naissance de l'infrarouge

Les rayons infrarouges bordent le spectre visible de la lumière blanche au dessous de rouge, chaque couleur de spectre a sa propre température, et en 1800 l'astronome anglais William Herschell a démontré une augmentation de la température de coté du rouge ou il n'y avait plus de lumière. Cette expérience montra que la chaleur peut se transmettre par une forme invisible de la lumière : infrarouge.

IV.2.1.2 Définition de l'infrarouge

La lumière visible est caractérisée par une plage de longueur d'onde allant de 400nm à 700nm. La lumière émise à 700nm est de couleur rouge, celle à 400 est de couleur violette et entre, nous retrouvons toutes les couleurs de l'arc en ciel. L'infrarouge comme son nom l'indique est la plage de longueur d'onde située au dessous du rouge. L'infrarouge n'est donc pas visible par l'œil humaine. L'infrarouge est un rayonnement électromagnétique invisible, de l'longueur d'onde comprise entre 0.8 µm (au dessous de la lumière rouge visible) et 1mm (micro-ondes). Les rayons proviennent souvent des sources de chaleurs, c'est d'ailleurs un système utilisé par l'armée pour repérer l'ennemi.

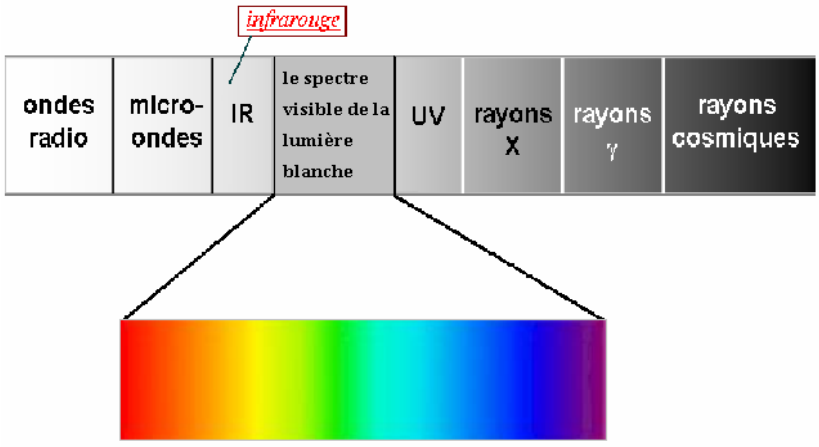


Figure IV.7 : le spectre électromagnétique

IV.2.1.3 Sources d'infrarouge

La première source de rayons infrarouge est le soleil .les radiations infrarouges de l'astre solaire traversent aisément l'atmosphère, même si certaines d'entre elles sont absorbées par le dioxyde de carbone et de vapeur d'eau parmi les autres sources infrarouges, on peut citer les lampes à incandescence à filaments de tungstène ou de carbone (lorsqu'elles sont en basse tension), certaines diodes, flammes, ou encore certain lasers à gaz.

Figure IV.8 : source d'infrarouge

IV.2.1.4 Détecteurs infrarouge

Un détecteur de rayonnement IR transforme ce rayonnement incident en un signal électrique. On distingue deux types de détecteurs :

- Les détecteurs thermiques qui ne sont sensibles qu'a l'énergie du rayonnement.
- Les détecteurs quantiques qui transforment les photons incidents en charges électriques.

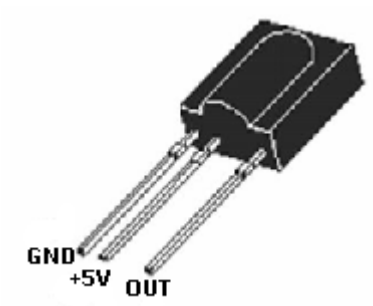


Figure IV.9 : capteur infrarouge

IV.2.1.5 Protocole de transmission RC5

Le code RC5, mis au point chez Philips, est devenu une norme pour les transmissions de commandes en infrarouge. C'est le code le plus employé pour les télécommandes infrarouge de TV, magnétoscope, chaînes HI FI, etc...

IV.2.1.5.1 Détails d'un bit en code RC5

Les bits du code RC5 sont codés en biphasé (codage Manchester), c'est à dire qu'un bit est composé de 2 demi-bits alternés. La durée d'un bit est constante : 1778 µs. Il y a toujours une transition au milieu du bit.

Un "0" se traduit par un niveau haut pendant 889 µs, suivi d'un niveau bas pendant 889 µs.

Un "1" se traduit par un niveau bas pendant 889 µs, suivi d'un niveau haut pendant 889 µs.

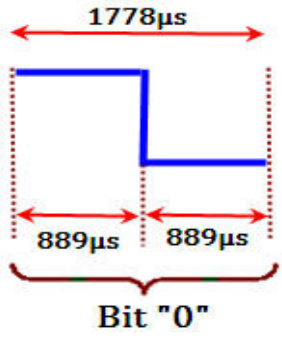


Figure IV.10 : bit 0 en biphasé

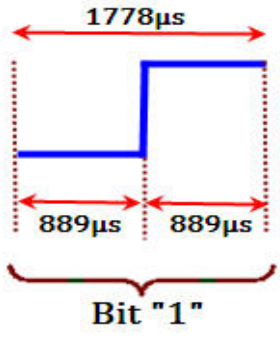


Figure IV.11 : bit 1 en biphasé

IV.2.1.5.2 Constitution d'une trame en code RC5

Elle est composée de 14 bits, qui sont, dans l'ordre de transmission :

- 2 bits toujours à "1" qui servent à la synchronisation.
- 1 bit de répétition, qui change d'état à chaque nouvel envoi de code.
- 5 bits d'adresse, pour la sélection de l'appareil à commander.
- 6 bits de code qui permettent de choisir la commande désirée.

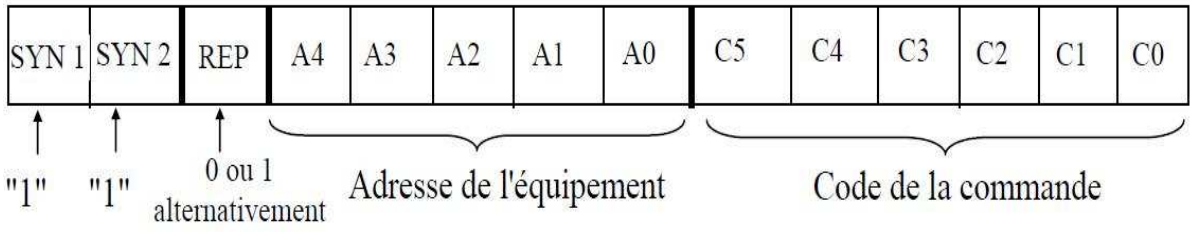


Figure IV.12 : Constitution d'une trame

- Les 2 bits de départ sont utiles pour ajuster le niveau de la commande automatique du gain AGC dans le circuit intégré de réception.
- Le bit de basculement indique une nouvelle transmission de données, sa valeur change à chaque nouvelle activation d'une touche afin de distinguer une nouvelle pression d'une pression continue sur la même touche.
- Les 5 bits suivants déterminent l'adresse du dispositif devant réagir à la commande. Nous avons donc 2^5 = 32 groupes d'adressage.
- L'instruction destinée à l'appareil est codée dans les 6 derniers bits ce qui donne 2^6 = 64 instructions.

Exemple de trame

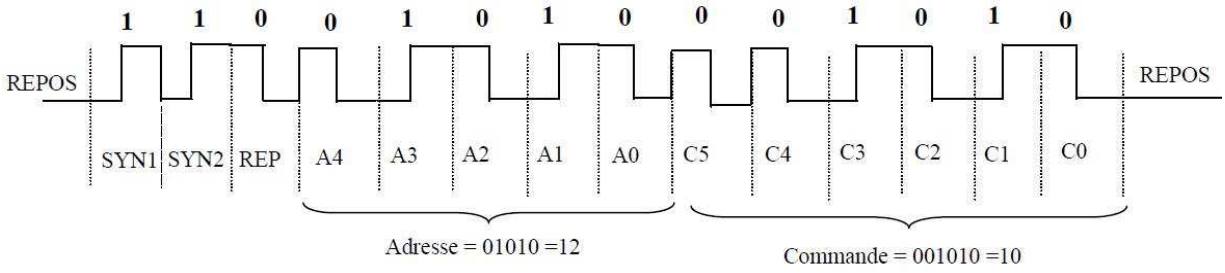


Figure IV.12 : exemple de trame

IV.2.1.5.2 Enchainement des trames

La trame dure : 14 x 1778 µs = 24,892 ms, la suivante ne sera émise que 89 ms après la fin de la précédente.

La périodicité des messages (ou trames) à été choisie comme étant un multiple de la durée d'un bit soit, 64 x 1,778 = 113,792 ms.

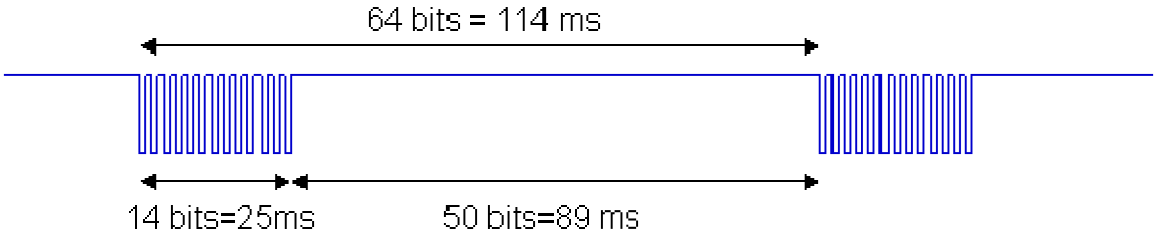


Figure IV.13 : enchainement des trames

IV.2.2 Principe et organigramme du codage de la trame

Pour que la télécommande puisse émettre les signaux (trames) associés aux 6 touches qu'elle contient, le microcontrôleur est doté d'un programme d'encodage, dont son organigramme est donné sur la page suivante, avec:

Code un : c'est une fonction qui génère un « 1 » en code biphasé (front montant).

Code zéro : c'est une fonction qui génère un « 0 » en code biphasé (front descendant).

Bit de synchronisation : c'est un « bit à zéro » qu'il faut envoyer à chaque début de la trame pour la synchronisation et pour le déclenchement de l'interruption sur RB0 ; Car, le « code_zéro » a un front descendant qui va être inversé par le récepteur infrarouge, donc on va avoir un front montant « bit à un », qui va déclencher l'interruption sur RB0.

j : pointeur de conteur.

V[j] : vecteur contenant les bits du signal de commande à émettre.

-Dans notre cas, on a envisagé d'utiliser une petite trame de 4 bits seulement, dont le premier bit sert à la synchronisation et déclenchement de l'interruption INT/RB0, et les 3 autres bits pour la commande, comme le montre la figure ci-dessous :

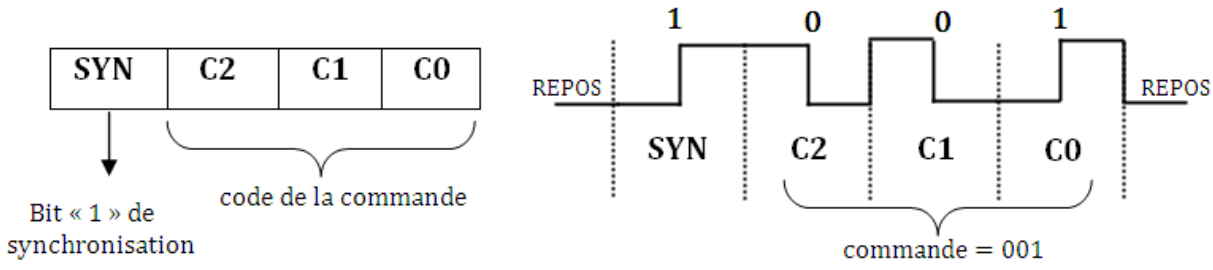


Figure IV.14 : constitution de notre trame + un exemple

-Pour chaque touche du clavier, on a associé un code de commande bien spécifié :

V1 [3]= {0,0,1}; (01 en Hexa) ;associé à la touche B0.
V2 [3]= {0,1,0}; (02 en Hexa) ;associé à la touche B1.
V3 [3]= {0,1,1}; (03 en Hexa) ;associé à la touche B2.
V4 [3]= {1,0,0}; (04 en Hexa) ;associé à la touche B3.
V5 [3]= {1,0,1}; (05 en Hexa) ;associé à la touche B4.
V6 [3]= {1,1,1}; (07 en Hexa) ;associé à la touche B5.

-La broche a0 est la sortie du PIC16F84A qui est utilisé pour l'émission série du code.

IV.2.2.1 Organigramme du codage de la trame

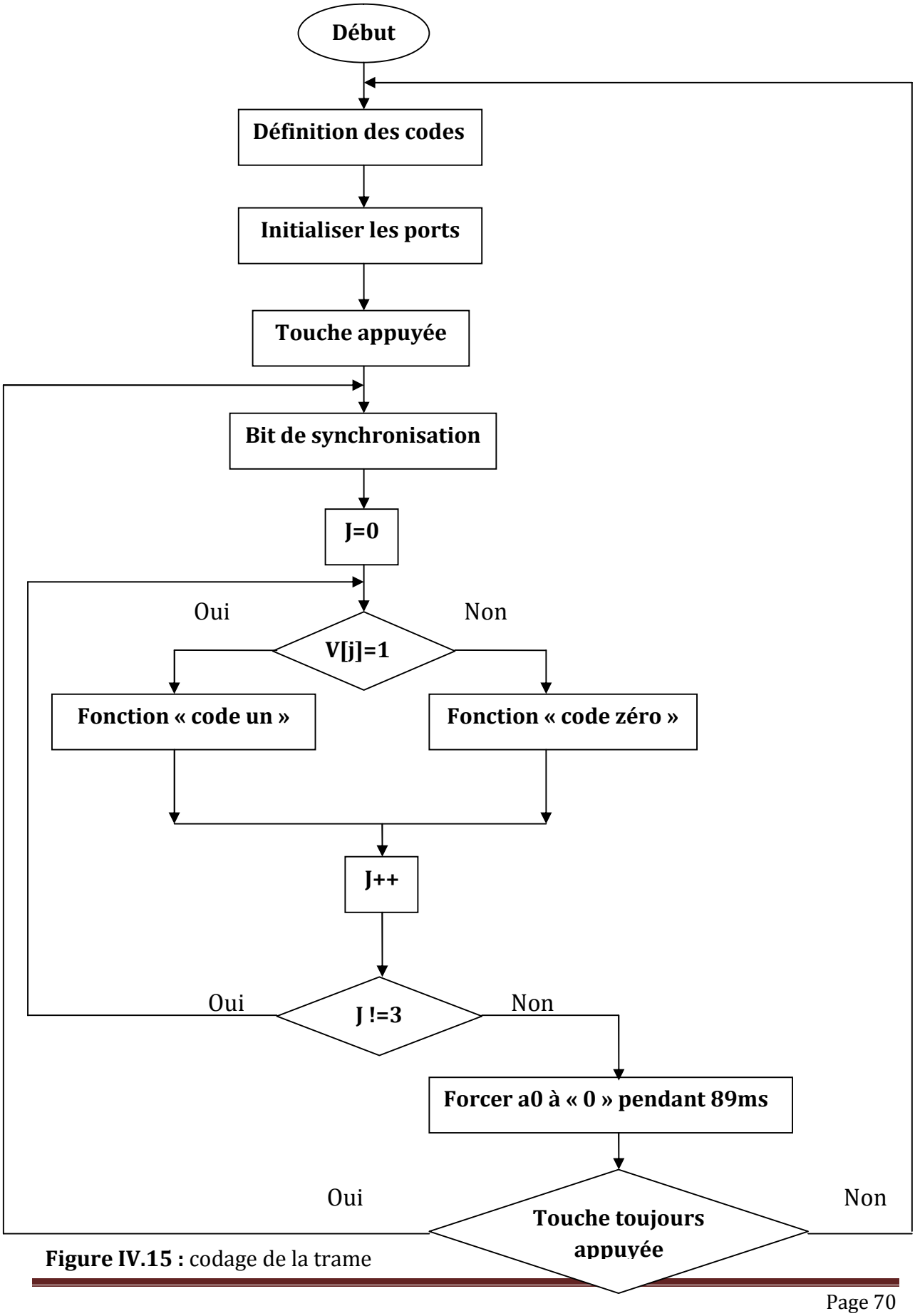


Figure IV.15 : codage de la trame

IV.2.2.1.1 Organigramme de la fonction `` code_un ''

Cette fonction permet de générer un bit « 1 » en biphasé, qui est traduit par un « 0 » pendant 889us suivi d'une fréquence de 38KHz pendant 889us, c'est un front montant qui va être inversé par le récepteur infrarouge (front descendant),

Pour générer la fréquence 38KHz (période 26us) pendant 889us, il faut boucler le programme (889/26=34).

Pour générer la fréquence 38KHz, il suffit forcer a0 à « 1 » pendant une demi-période (13us), suivi de forçage de a0 à « 0 » pendant l'autre demi-période (13us).

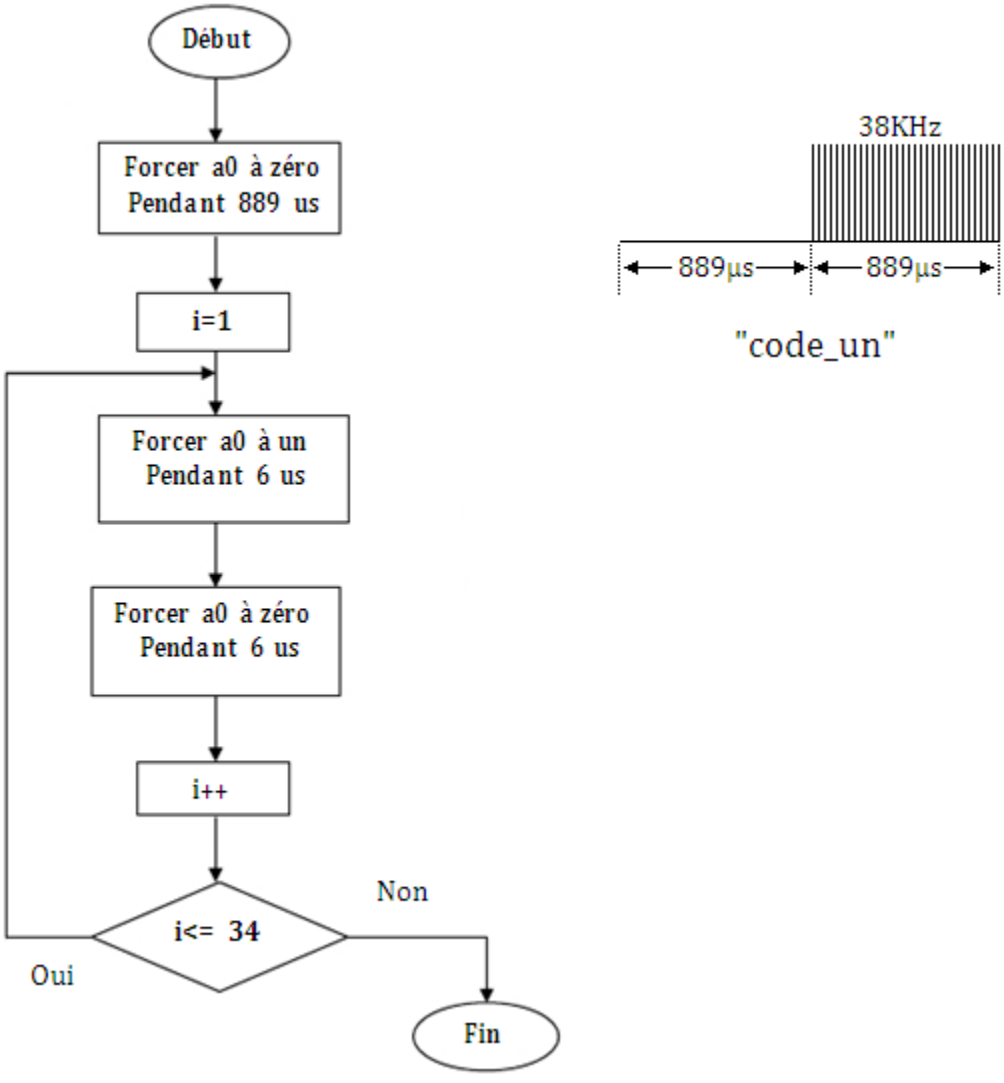


Figure IV.16 : Organigramme de la fonction `` code_un''

IV.2.2.1.2 Organigramme de la fonction `` code_zéro ''

Cette fonction permet de générer un bit « 0 » en biphasé, qui est traduit par une fréquence de 38KHz pendant 889us suivi d'un « 0 » pendant 889us, c'est un front descendant qui va être inversé par le récepteur infrarouge (front montant).

Pour la génération de fréquence 38KHz, c'est exactement la même chose à celle de la fonction "code_un".

Il faut savoir que le capteur infrarouge est sensible seulement à la présence ou l'absence de la fréquence.

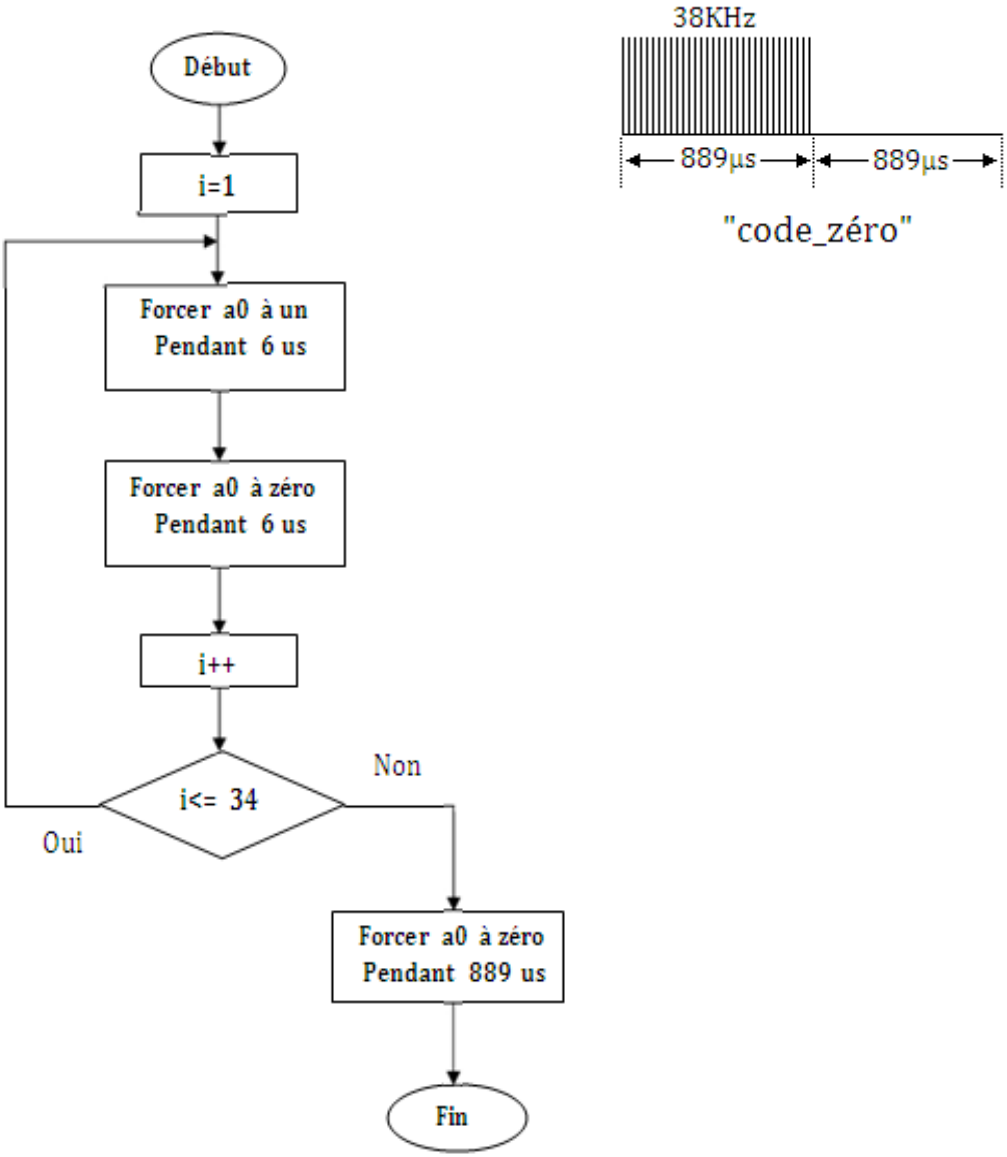
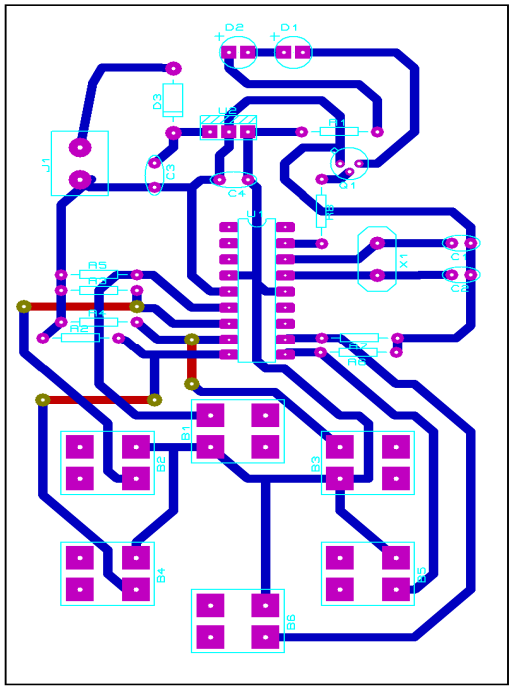


Figure IV.17 : Organigramme de la fonction `` code_zéro''

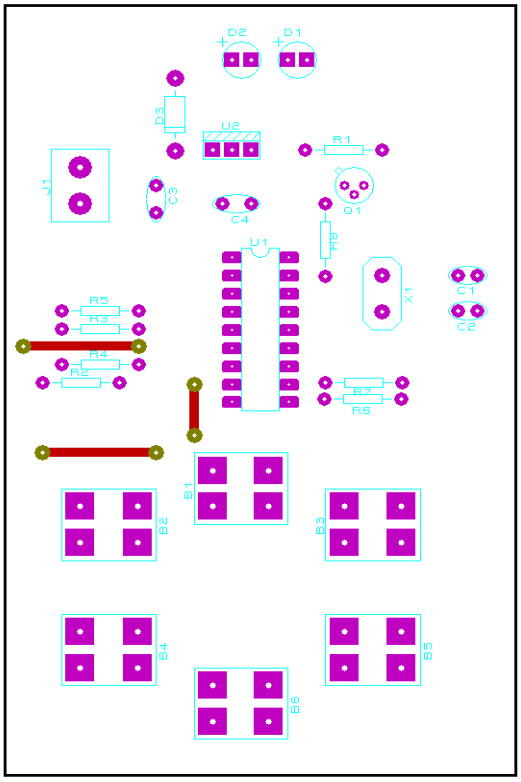
IV.2.3 Réalisation Pratique :

IV.2.3.1 Le typon de carte de la télécommande

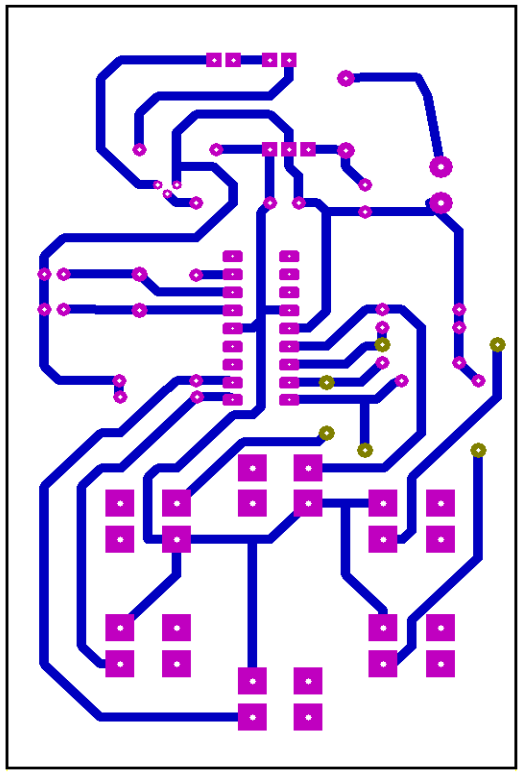


IV.2.3.2 Le circuit imprimé de la télécommande

- Coté composants



Coté cuivre :



IV.3 Conclusion

Dans ce chapitre nous avons présenté la partie matérielle de la télécommande de pilotage du robot avec ses différents blocs (alimentation, clavier, microcontrôleur, bloc d'émission), ainsi, les différents organigrammes qui décrivent le fonctionnement de notre télécommande infrarouge.

CHAPITRE V

LE ROBOT RÉALISÉ

Carte De Commande

V.1 Introduction

Dans ce chapitre, nous décrivons les différentes parties réalisées pour la mise en œuvre du notre robot. D'abord, nous commençons par une brève description du noyau du robot qui sert à commander selon les différents étages électroniques ; puis nous explicitons la partie appropriée à la perception de l'environnement. En fin, on donne une description des circuits de commande en puissance des deux moteurs à courant continu, ainsi que du moteur pas à pas bipolaire.

V.2 Le principe général de fonctionnement

Les différentes parties de la carte de commande, sont présentées sur le schéma suivant :

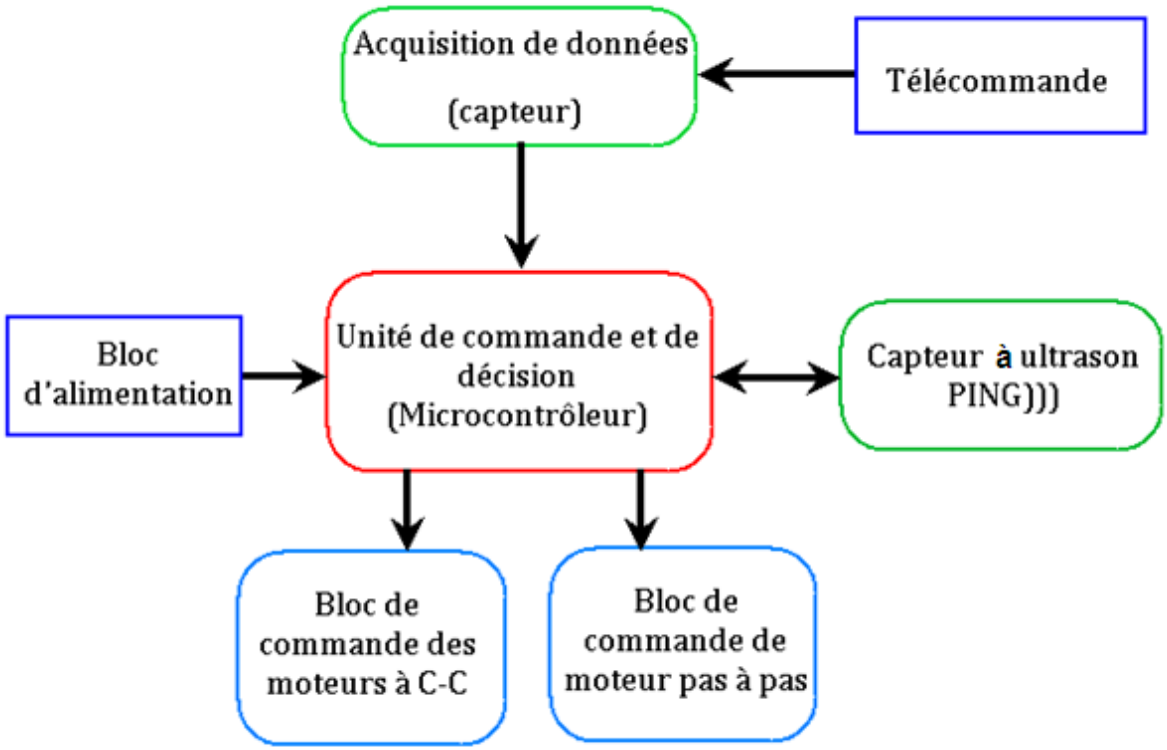


Figure V.1 : Schéma synoptique du robot réalisé

A partir de ce schéma, on peut synthétiser les différents blocs de notre carte de commande qui sont:

- Bloc d'alimentation.
- Unité de commande et de décision qui est le PIC 16F877A.
- Bloc de commande des moteurs à courant continu.
- Bloc de commande du moteur pas à pas bipolaire.
- Partie acquisition de données.
- Partie système ultrasonique PING))).

V.2.1 Bloc d'alimentation

Nous avons utilisé deux blocs d'alimentation séparée :

Le premier bloc d'alimentation est constitué de deux alimentations nécessaires au fonctionnement des circuits de la carte de commande à savoir : l'alimentation de puissance de 12volts pour alimenter le moteur pas à pas bipolaire, et l'alimentation de 5 volts pour les circuits logiques.

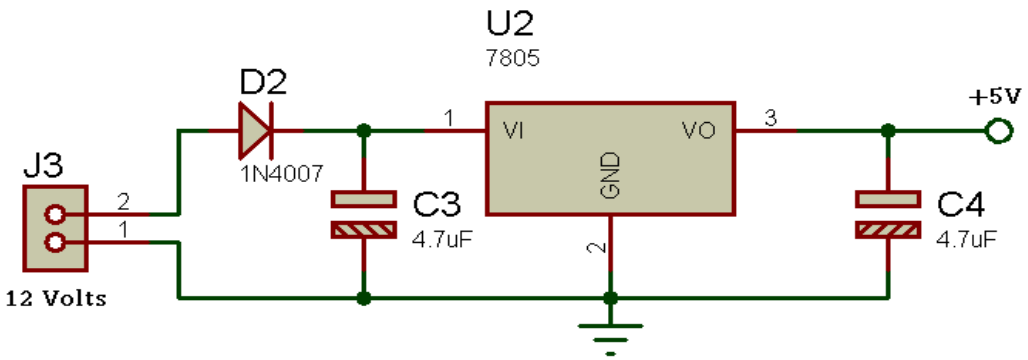


Figure V.2 : Le premier bloc d'alimentation

L'alimentation logique est obtenue à partir de l'alimentation de puissance 12volts, réglée par le régulateur de tension LM7805.

Pour filtrer les bruits (parasites), on a placé deux condensateurs polarisés (C3 et C4). Quant à la diode (D2) est utilisée pour éviter les chocs électriques dus à une inversion de branchement de la batterie.

Le deuxième bloc d'alimentation est constitué d'une alimentation de puissance de 9 volts pour alimenter les deux moteurs à courant continu, comme le montre la figure suivante :

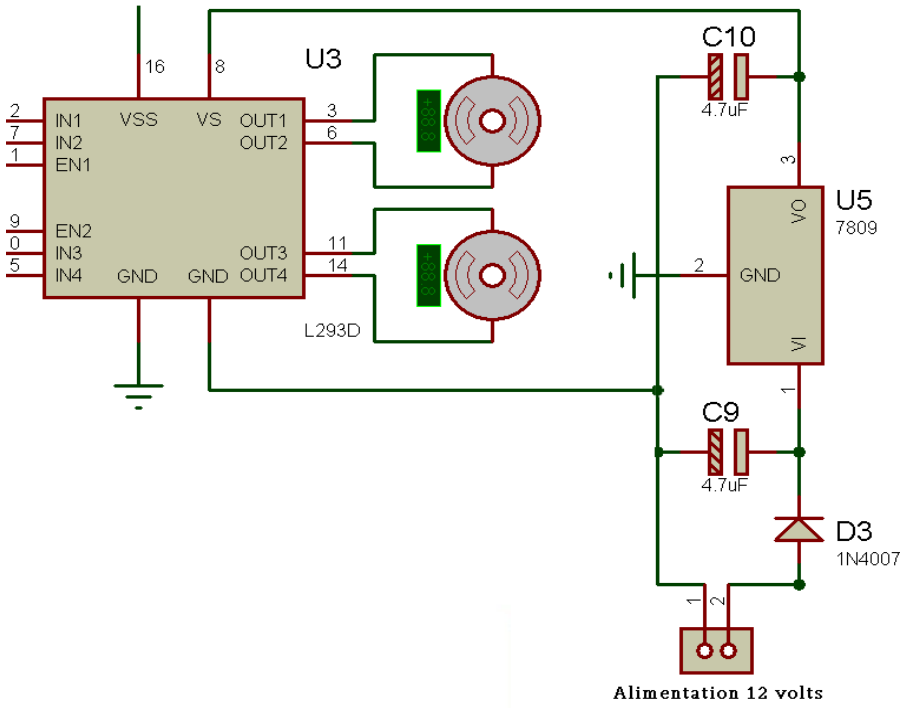


Figure V.3 : Circuit d'alimentation des deux moteurs à courant continu

L'alimentation de puissance de 9 volts est obtenue à partir de l'alimentation 12 volts, réglée par le régulateur de tension LM7809.

V.2.2 Unité de commande et de décision

Elle se compose d'un microcontrôleur et ses circuits annexes (oscillateur, reset, alimentation et sélectionneur de tâche).

V.2.2.1 Le microcontrôleur PIC 16F877A:

C'est le cerveau de la carte de commande, qui génère des requêtes appropries en fonction des signaux collectés. Son fonctionnement et ses caractéristiques sont étudient dans le chapitre (II).

C'est le noyau du robot qui sert à commander et lier les différents étages électroniques.

V.2.2.2 Le circuit reset :

Ce circuit va permettre de provoquer un reset (réinitialisation) à chaque fois que la broche /MCLR reçoit un « 0 » (la mise à la masse), par l'appui sur le bouton poussoir.

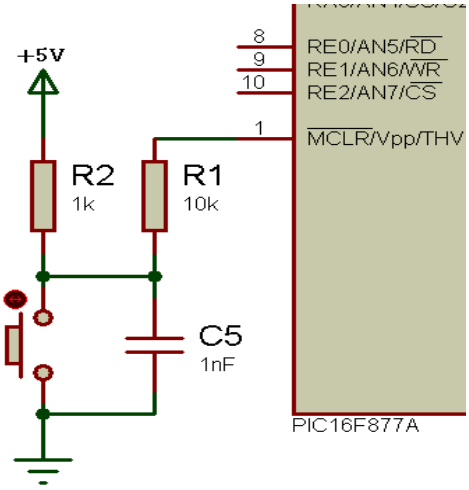


Figure V.4 : Circuit reset

V.2.2.3 L'oscillateur :

Pour que le microcontrôleur fonctionne correctement, il est nécessaire d'utiliser un signal d'horloge, dont le rôle est de cadencer tous les échanges soit internes (de registre vers registre), soit externes (de registre vers un port d'entrées-sorties).

Il existe plusieurs oscillateurs : LP : quartz de faible puissance.

XT : quartz externe.

HS : quartz externe rapide.

RC : circuit RC externe.

Pour notre carte, ce signal d'horloge est généré par le circuit schématisé dans la figure suivante :

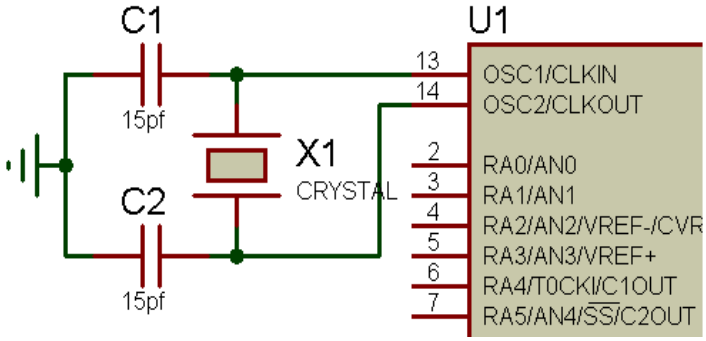
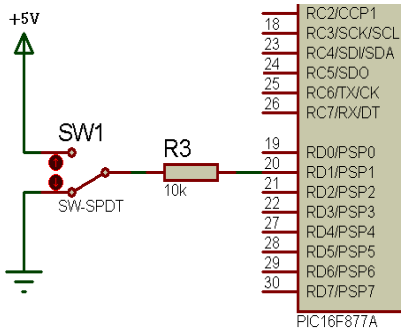


Figure V.5 : Circuit d'horloge

Dans ce circuit, on a utilisé un quartz de 4MHz associé au deux condensateurs C1 et C2. Ce quartz de 4MHz permet de cadencer le microcontrôleur à une fréquence de 1MHz, ce qui fait un million d'instruction par seconde (1µs).

V.2.2.4 Circuit sélectionneur de tâches

Ce circuit représente l'interface homme-robot. En effet, grâce au commutateur de ce circuit, (représenter sur la figure suivante), l'operateur peut sélectionner la tâche à réaliser. La sélection de tâche se fait comme suit : (Verrouillage des tâches)



SW1	La tâche à exécuter	Détails
« 0 » GND	Tâche (1)	Evitement d'obstacles
« 1 » +5V	Tâche (2)	Téléguidage par télécommande

Figure V.6 : Circuit sélectionneur de tâches

V.2.3 Bloc acquisition de données

Pour commander notre robot à distance, on a besoin d'utiliser le capteur de signal infrarouge TSOP 1738 ; qui se charge de collecter les salves quand le signal est envoyé par la télécommande, ensuite le traiter au niveau de microcontrôleur PIC.

Le branchement de TSOP1738 au microcontrôleur PIC est représenté sur cette figure :

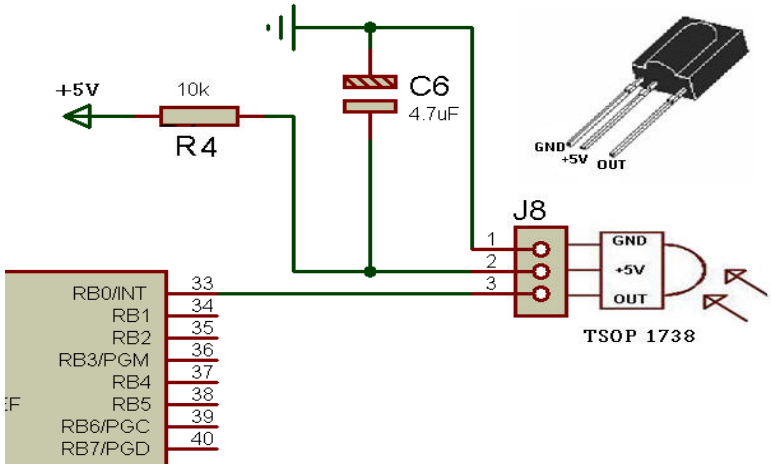


Figure V.7 : Branchement du récepteur infrarouge TSOP 1738 au PIC

V.2.3.1 Le capteur TSOP1738 :

Le TSOP 1738 se charge de collecter les trames envoyées par l'émetteur puis les transforment en charges électriques.

V.2.3.2 Branchement du TSOP 1738 :

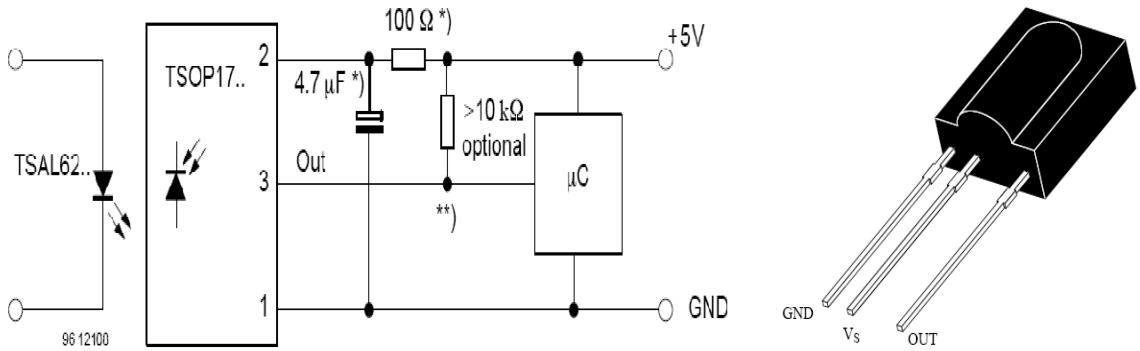


Figure V.8 : Capteur infrarouge TSOP 1738

Ce récepteur réagit à un faisceau infrarouge modulé à une fréquence de 38khz.
Au repos sa sortie est au niveau 1, elle passe au niveau 0 lors de la réception.

V.2.3.3 Schéma et principe de fonctionnement du TSOP 1738

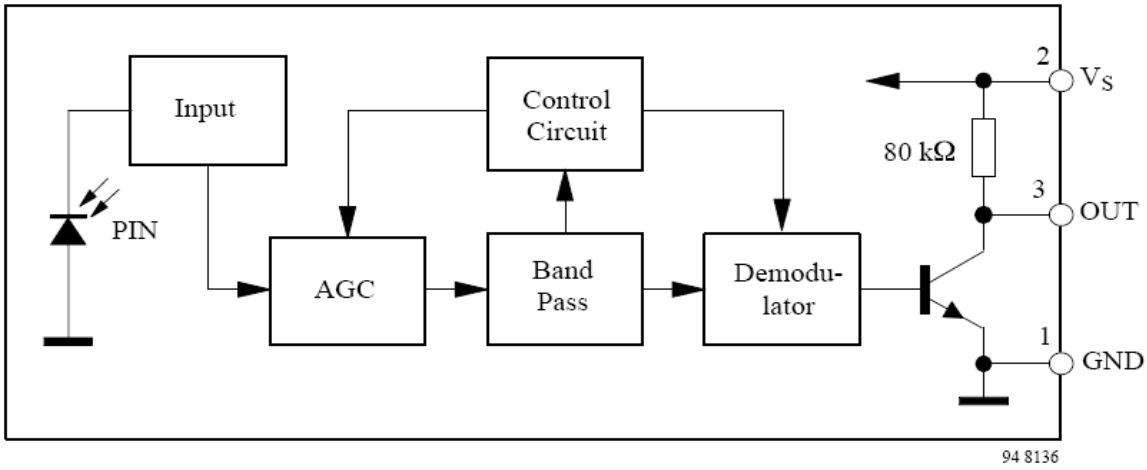


Figure V.9 : Schéma de principe de TSOP 1738

D'après son schéma de principe donné par la figure ci-dessus, on constate que les capteurs infrarouges effectuent la réception, l'amplification, la démodulation du signal. Ils sont fabriqués pour fonctionner sur des fréquences se situant au voisinage du 38khz. Chaque fabricant à ses propres brochages et sa fréquence optimale précise, mais tous ont trois broches : la masse (GND), l'alimentation (+5V) et la sortie du signal (OUT).

V.2.4 Partie système ultrasonique PING)))

Après le branchement de l'alimentation (+5v) et (GND) du capteur à ultrason PING)) ; on a utilisé une seule broche du microcontrôleur PIC, pour communiquer avec ce capteur. Pour le capteur PING)) : Son fonctionnement et ses caractéristiques sont étudiées dans le chapitre (III).

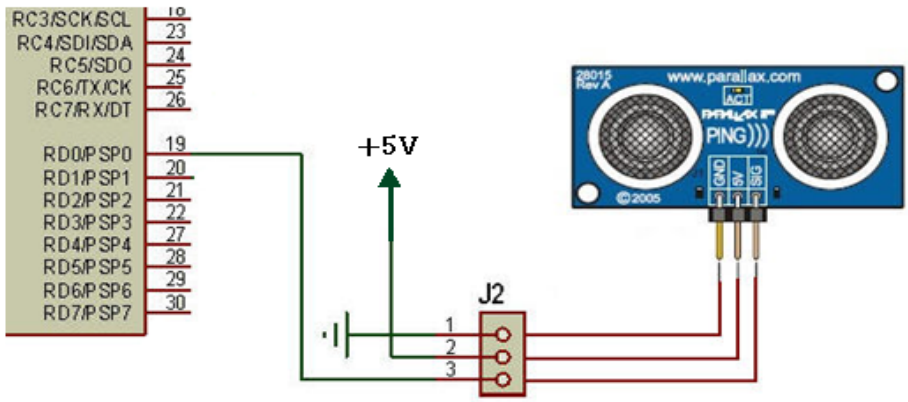


Figure V.10 : Branchement du capteur à ultrason au microcontrôleur PIC

V.2.5 Bloc de commande des moteurs à courant continu

Pour que le robot puisse se déplacer, deux moteurs à courant continu de faible puissance sont utilisés. L'un pour entraîner les roues motrices, et l'autre pour entraîner les roues directrices.

V.2.5.1 Principe d'inversion de sens d'un moteur à courant continu

La première idée qui vient à l'esprit lorsqu'on veut inverser les polarités d'un moteur à courant continu est le schéma suivant :

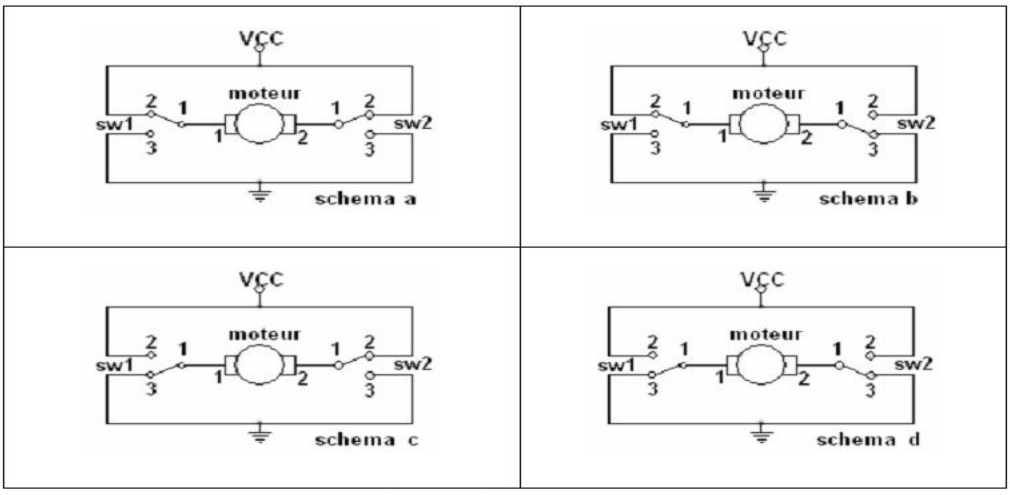


Figure V.11: Principe d'inversion de sens d'un moteur à courant continu

En regardant les schémas précédents, on peut déduire le sens de rotation du moteur :

Sur le schéma (a), le moteur est à l'arrêt (on peut dire même qu'il est freiné : en effet court-circuiter les deux pôles d'un moteur revient à le freiner.

Sur le schéma (b), il tourne dans le sens inverse du schéma (c). Et enfin dans le schéma (d), il est freiné. C'est la base du « pont en H », toute l'idée réside dans ce schéma.

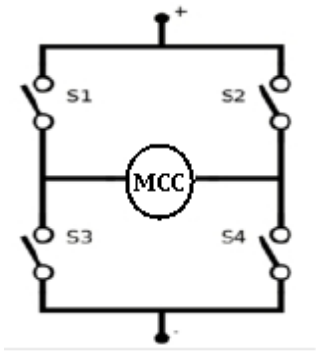


Figure V.12: Principe de pont de H.

Dans notre cas, on a deux moteurs à courant continu, donc on a besoin de deux circuits « pont en H » pour pouvoir les commander.

On a envisagé d'utiliser un circuit intégré qui remplace nos deux circuits « pont en H » pour des raisons de simplicité et aussi pour gagner plus d'espace.

Le circuit utiliser est le fameux « L293D ». Donc, le circuit de commande des deux moteurs à courant continu sera le suivant :

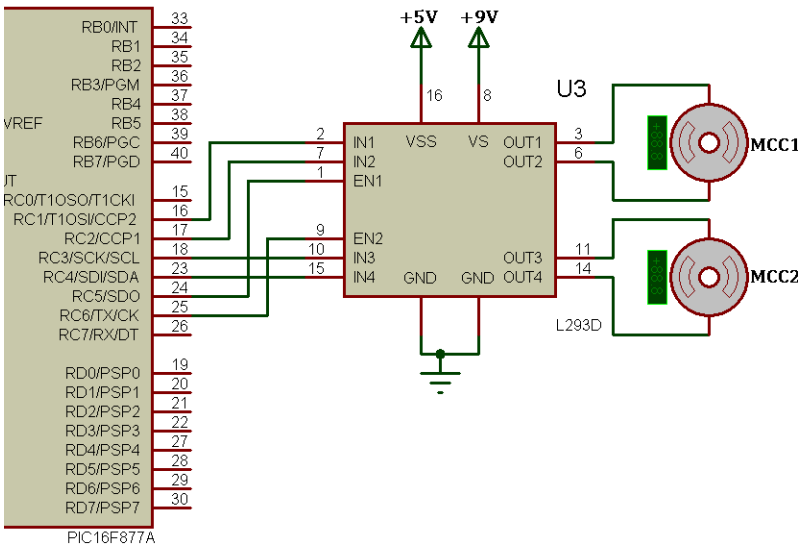
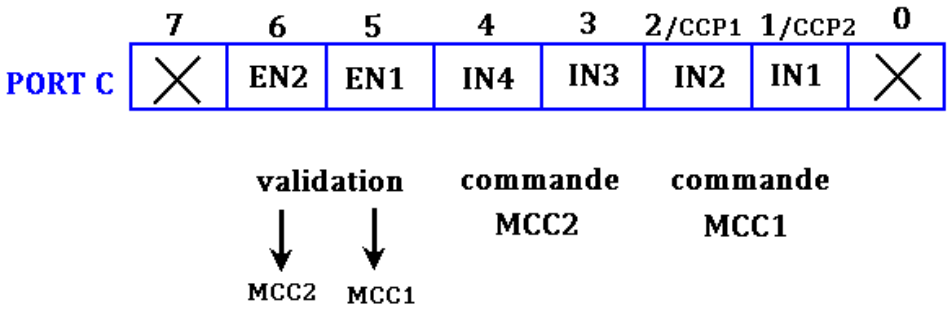


Figure V.13: Circuit de commande des deux moteurs à courant continu

Pour commander les deux moteurs par le microcontrôleur PIC, nous avons relié les entrées de commande de circuit intégré L293D au « PORT C » du PIC, comme le montre la figure précédente ; et pour avoir les différentes manœuvres sur le chariot, nous avons procédé comme suit :



Alors, les deux broches RC1 et RC2 qui sont reliées respectivement aux bits de commande IN1 et IN2 qui servent à commander le moteur MCC1 dans les deux sens.

Même chose pour commander le moteur MCC2, on a utilisé les deux broches RC3 et RC4 qui sont reliées respectivement aux bits de commande IN3 et IN4. On remarque bien que les broches 1 et 2 du PORT C ont deux fonctions, qu'on peut les utilisés soit comme sorties de commande simple ou bien comme sorties de CCP, activant le mode PWM (MLI) pour contrôler la vitesse de rotation du MCC1 dans les deux sens. Enfin, les deux broches RC5 et RC6, qui sont reliées respectivement aux bits de commande EN1 et EN2 servent à valider l'un des moteur ou les deux au même temps par le niveau haut.

Connaissant le fonctionnement du circuit L293D et son implantation sur la carte, nous avons retiré les codes relatifs à chaque manœuvre :

-marche avant : PORTC=0b00100100=0x24.

-marche arrière : PORTC=0b00100010=0x22.

-tourner à droite en avant : PORTC=0b01101100=0x6c.

-tourner à droite en arrière : PORTC= 0b01101010=0x6a.

-tourner à gauche en avant : PORTC=0b01110100=0x74.

-tourner à gauche en arrière : PORTC=0b01110010=0x72.

V.2.6 Bloc de commande du moteur pas à pas bipolaire

Le moteur pas à pas utilisé pour orienter le capteur d'ultrason PING)) dans les différentes directions afin de mesurer la distance des obstacles qui entour le robot ; ce moteur est de type bipolaire de faible puissance, ce qui veut dire qu'il a deux bobines, alors à quatre bornes. La bonne polarisation de ses deux bobines, nous assure la rotation du moteur dans un sens ou dans un autre. Dans ce cas on a besoin de deux circuits « pont en H » pour pouvoir le commander. Nous avons envisagé d'utiliser aussi un deuxième circuit intégré « L293D» qui remplace les deux « ponts en H » pour des raisons de simplicité et aussi pour gagner plus d'espace. Donc le circuit de commande du moteur pas à pas bipolaire utilisé est le suivant :

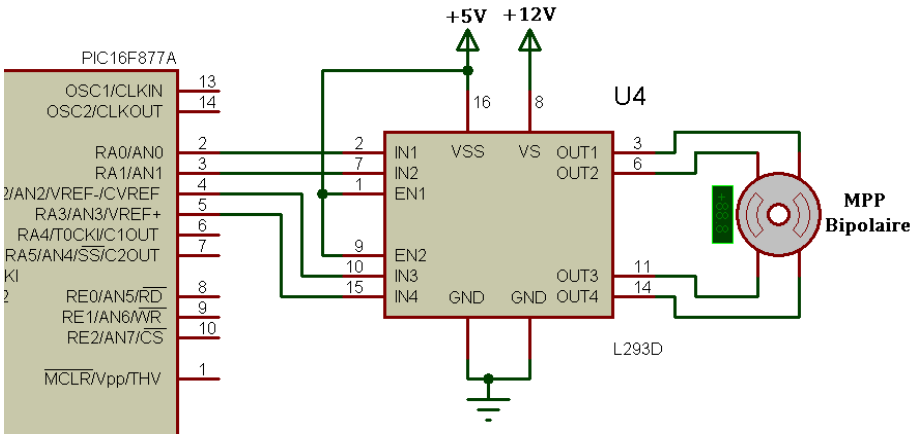
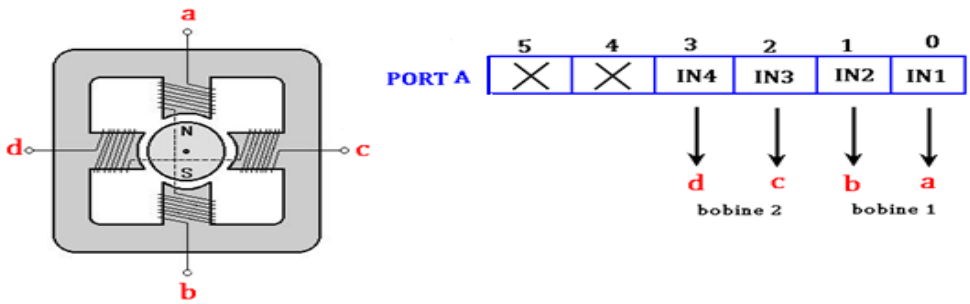


Figure V.14: Circuit de commande du moteur pas à pas bipolaire

V.2.6.1 La séquence de commande du moteur pas à pas utilisé

Pour commander le moteur pas à pas, par le microcontrôleur PIC, nous avons relié les entrées de commande du circuit intégré L293D (IN1, IN2, IN3, IN4) au « PORT A » du PIC comme le montre la figure suivante ; et pour avoir les différentes rotations à droite et à gauche du moteur, nous avons procéder comme suit :



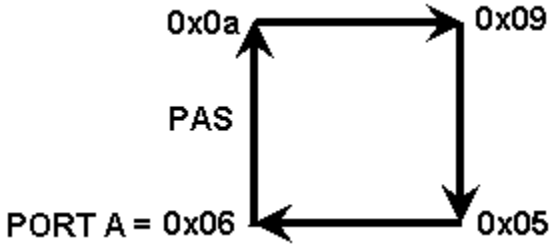
On a relié la bobine 1(a,b) aux broches de sorties OUT1, OUT2 de circuit intégré L293D, et les deux broches de commande IN1, IN2 sont connectées aux broches RA0 ,RA1 de PORT A du PIC .

Quant à la bobine 2(c, d) est relié aux broches de sorties OUT3, OUT4 de circuit intégré L293D, et les deux broches de commande IN3, IN4 sont connectées aux broches RA2, RA3 de PORT A du PIC.

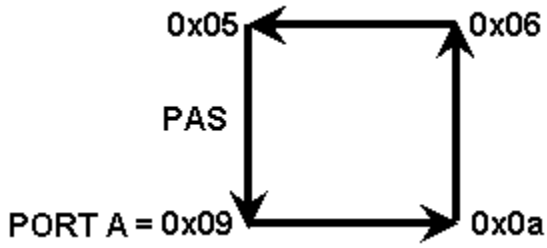
Les deux autres broches EN1 et EN2 sont reliées directement aux +5V (état haut) pour valider les deux étages «pont en H » du circuit L293D.

Le sens de rotation du moteur pas à pas bipolaire dépend du sens de courant et de l'ordre d'alimentation des bobinages ; donc, il faut avoir des séquences bien spécifiées pour avoir la rotation désiré.

A- La séquence de commande pour le tourner à droite :



B- La séquence de commande pour le tourner à gauche :



V.2.7 la carte électronique de commande du robot

Le schéma électrique de la carte de commande est donné par cette figure :

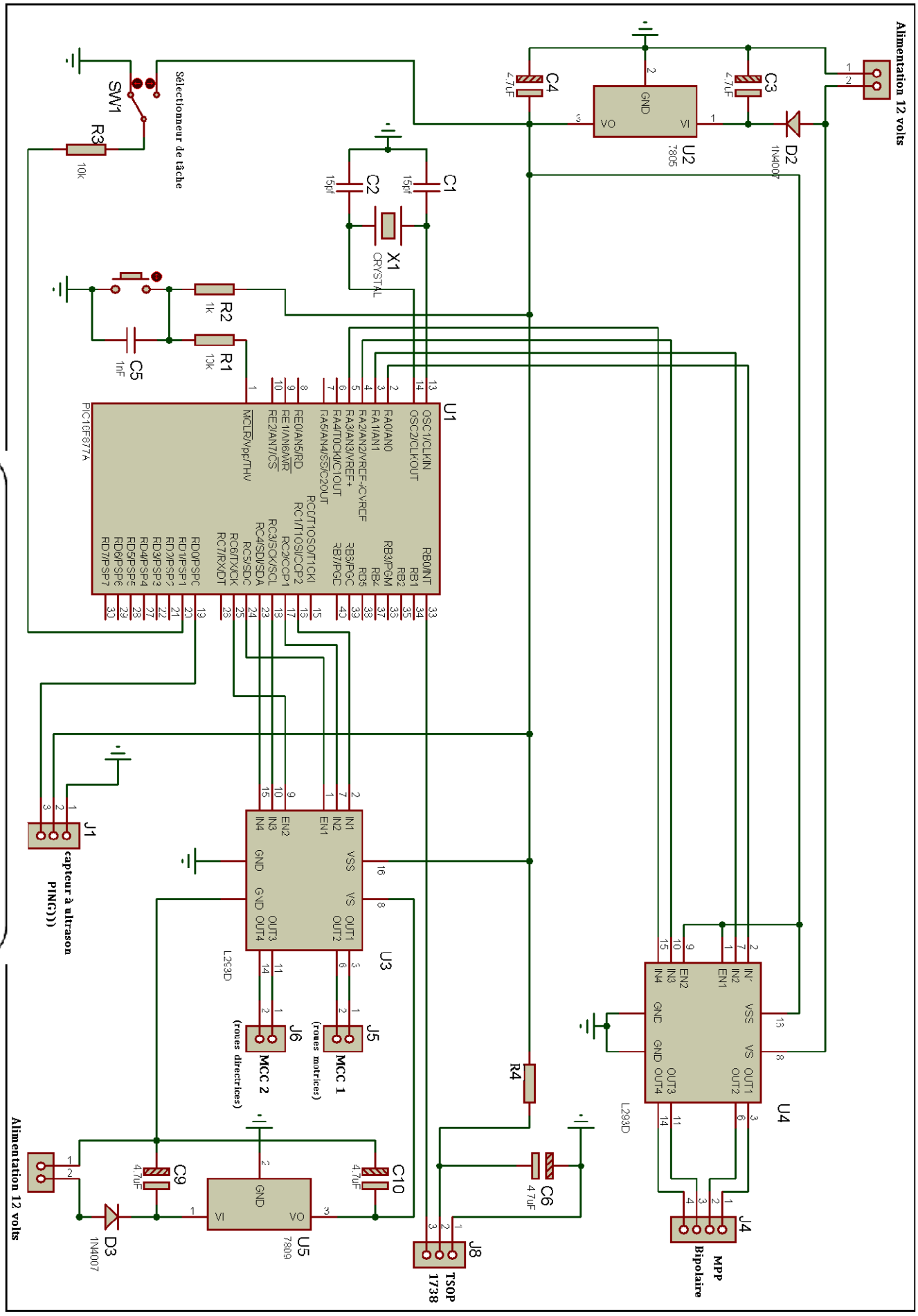


Figure V.15 : Le circuit de commande du Robot

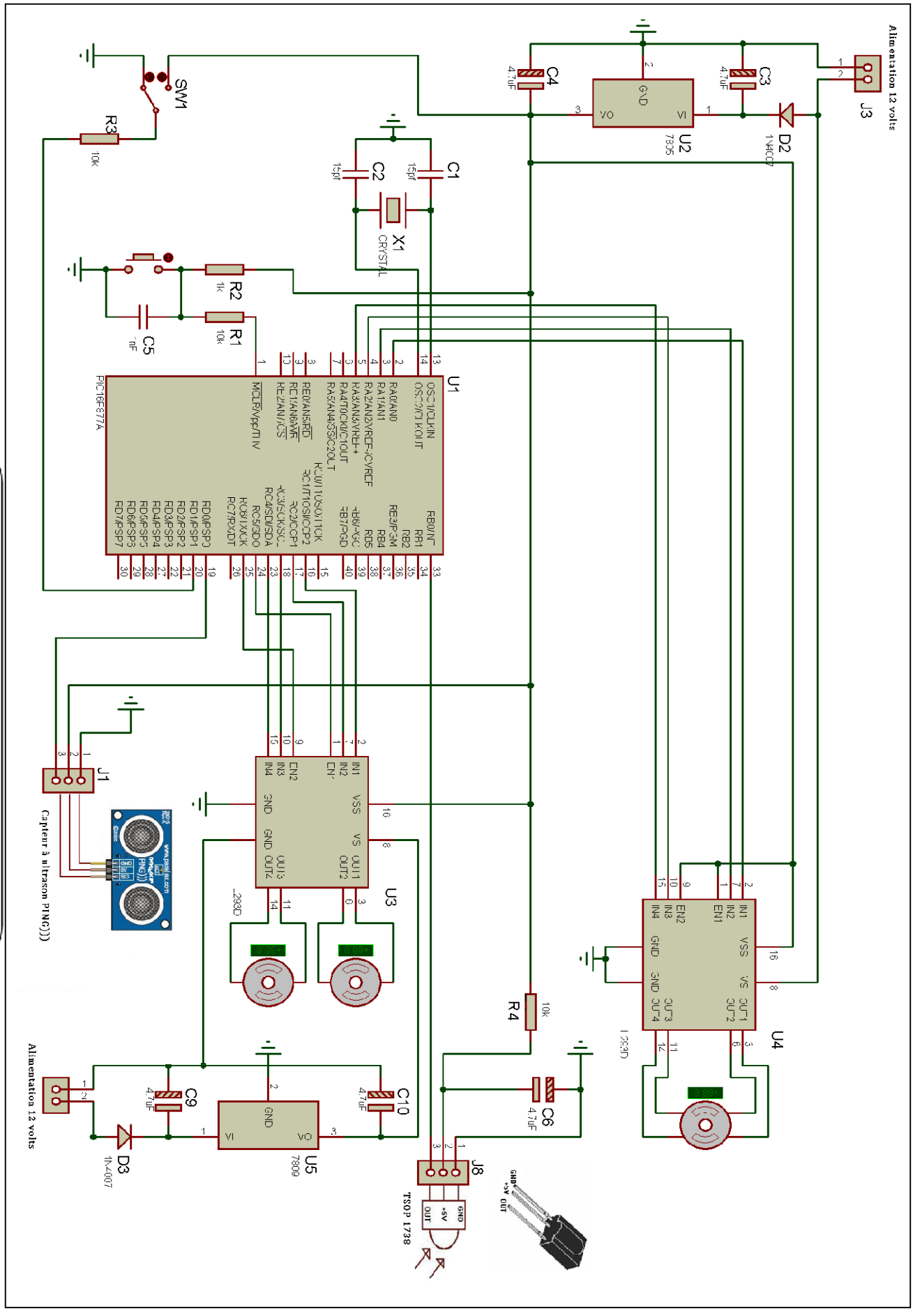


Figure V.15 : Le circuit de commande du Robot

V.3 conception logicielle de la carte de commande

Le microcontrôleur est un circuit programmable, et comme il est le cerveau de notre robot, il doit être programmé pour gérer la partie hardware qui regroupe l'ensemble des circuits des moteurs, et les systèmes de perception : infrarouge et ultrason utilisés.

Cette partie est consacrée à la partie programmation du PIC, où on va présenter les différents organigrammes pour que le microcontrôleur puisse gérer les différents circuits, et commander la partie mécanique du robot.

V.3.1 Stratégie de navigation

Comme nous l'avons déjà déclaré dans le cahier de charge, le robot doit accomplir deux tâches distinctes à sélection manuelle, verrouillées par un interrupteur comme suit :

-Si **SW** relié à **GNG** ==> RD1=**0**, la tâche (1) sélectionnée (évitement d'obstacles).

-Si **SW** relié à **+5V** ==> RD1=**1**, la tâche (2) sélectionnée (Téléguidage par télécommande).

V.3.2 Organigramme de fonctionnement du robot

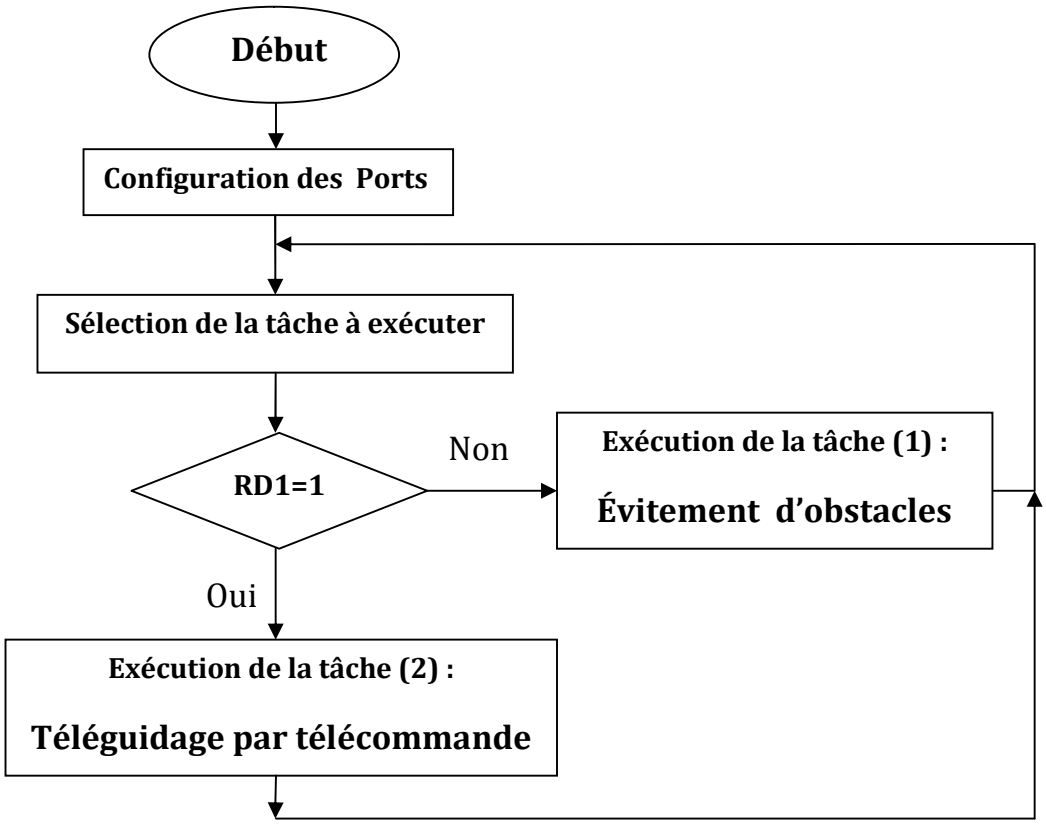
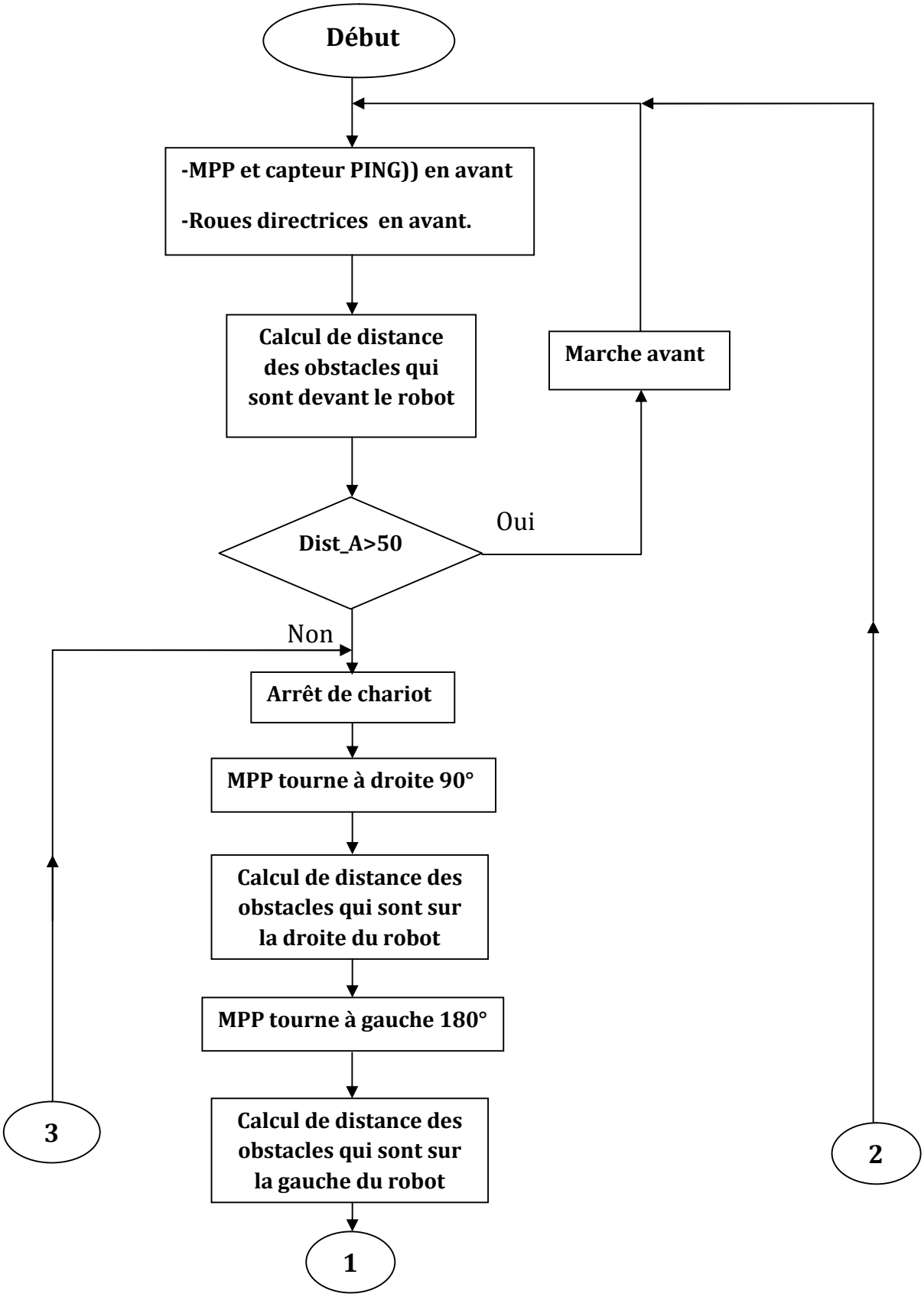


Figure V.16 : Organigramme de fonctionnement du robot

V.3. 3 Organigramme de programme principal de la tâche (1)



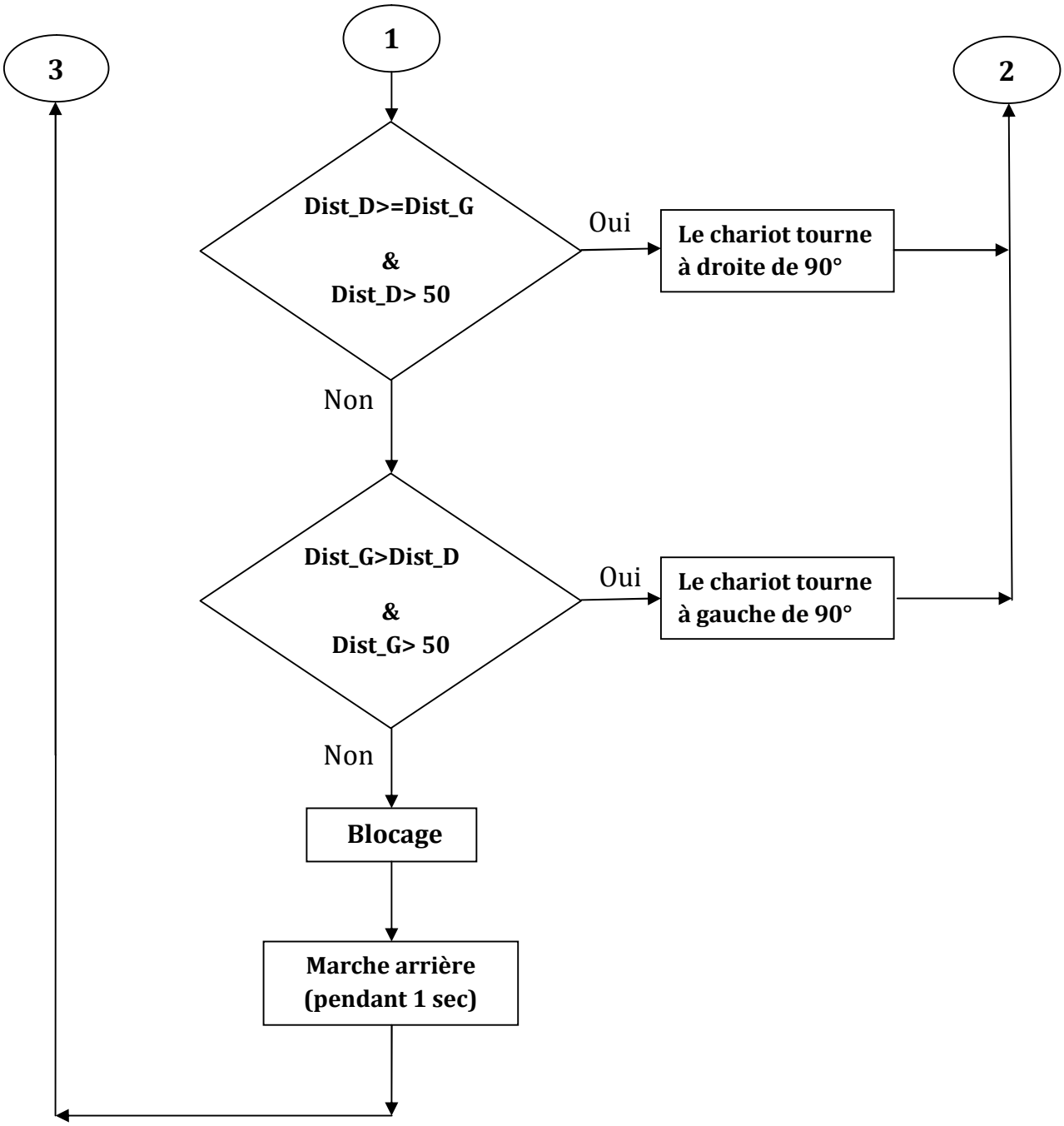
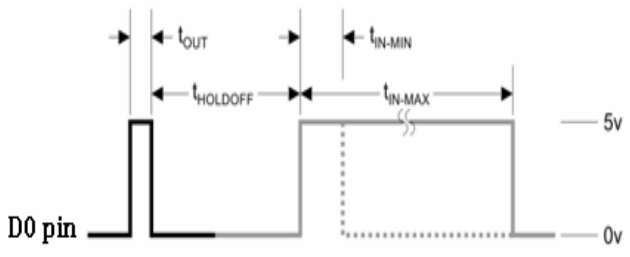


Figure V.17 : Organigramme de la tâche (1)

V.3. 4 Les fonctions de la tache (1)

La première fonction à présenter est celle de calcul de distance en utilisant le protocole de communication avec le capteur à ultrason PING))).



V.3.4.1 Organigramme de la fonction “Calcul_Distance“

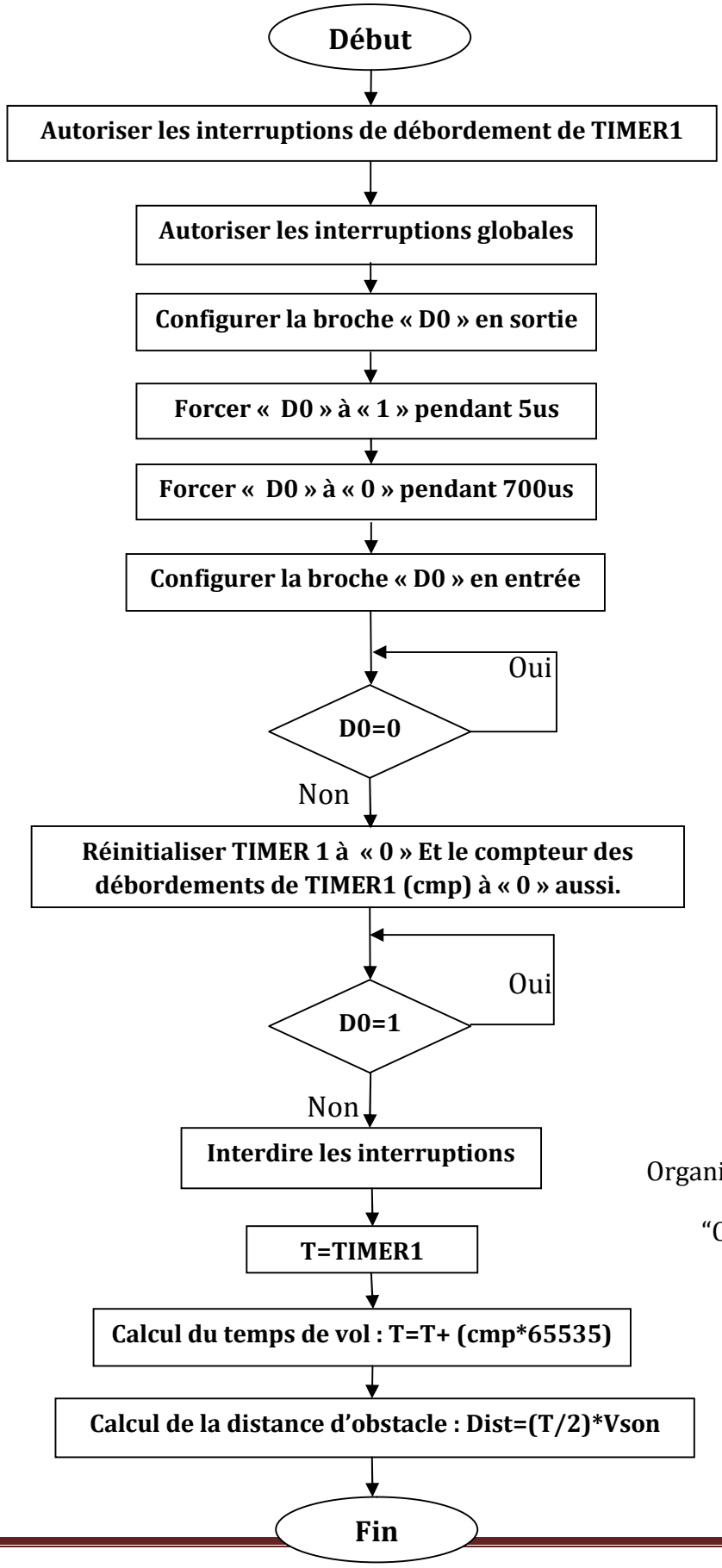


Figure V.18 :

Organigramme de la fonction

“Calcul_Distance“

V.3.4.2 Organigramme de la fonction "MPP_tourne_à_droite_90°"

Notre moteur pas à pas est un moteur bipolaire à 48 pas, c.à.d. chaque pas est de 360°/48 = 7.5° (pas = un angle de 7.5 dégrée).D'abord, il faut charger le vecteur des séquences associé au moteur pour tourner à droite. Pour que le moteur tourne à droite de 90°, il faut faire 90°/7.5°=12 fois le pas c'est ce que nous avons procéder dans l'organigramme suivant :

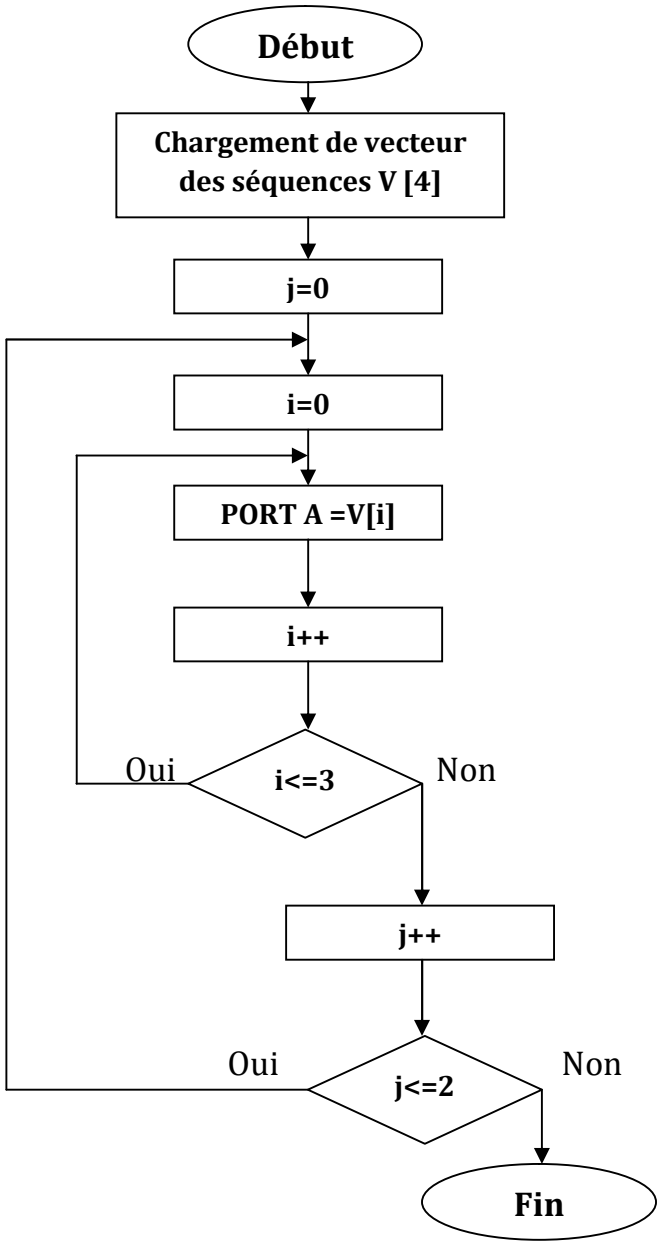


Figure V.19 : Organigramme de la fonction "MPP_tourne_à_droite_90°"

Remarque : le vecteur des séquences pour tourner à droite est :

V [4]= {0x06, 0x0a, 0x09, 0x05};

V.3.4.3 Organigramme de la fonction "MPP_tourne_à_gauche_180°"

C'est la même chose que le cas précédent, le pas est toujours de 7.5° mais le vecteur des séquences n'est pas le même .Alors, pour qu'il tourne à gauche de 180°, il faut faire 180°/7.5°=24 fois le pas c'est ce que nous avons procéder dans l'organigramme suivant :

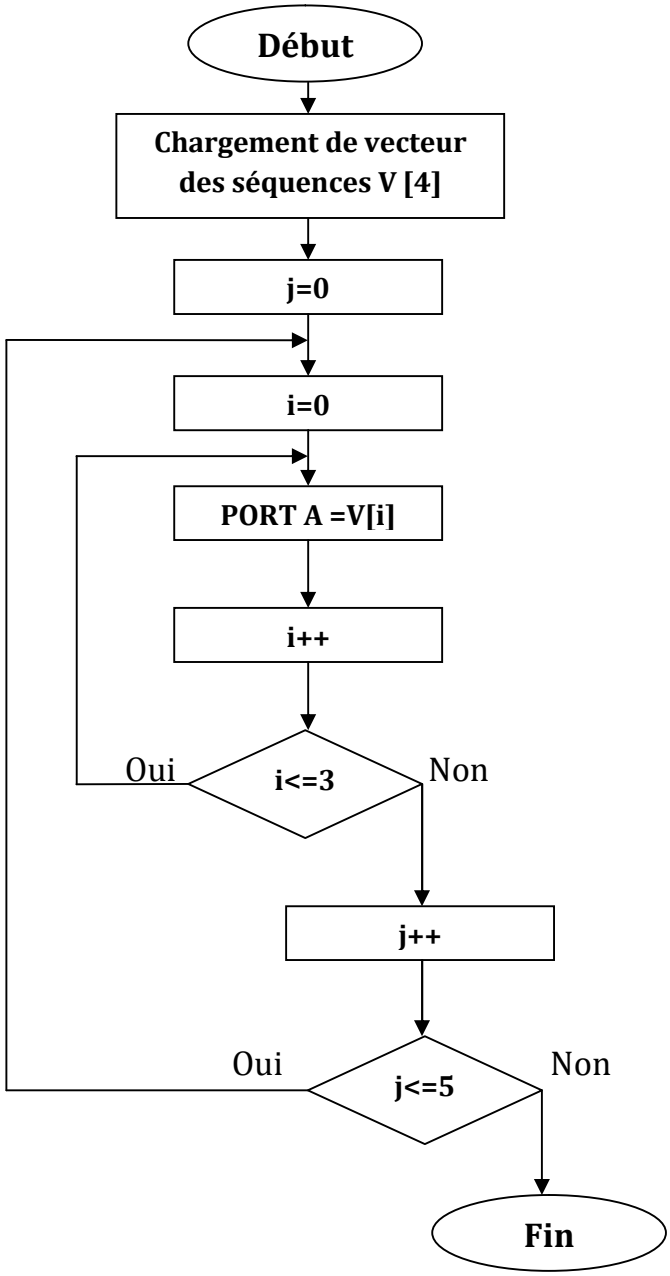


Figure V.20 : Organigramme de la fonction "MPP_tourne_à_gauche_180°"

Remarque : le vecteur des séquences pour tourner à gauche est :

V [4] = {0x09, 0x0a, 0x06, 0x05};

V.3.4.4 Les fonctions Marche avant et marche arrière

Pour ces deux manœuvres, il suffit que le microcontrôleur envoie le code associé sur le PORT C, auquel est relié le circuit de commande des moteurs (L293D).

Dans ces deux manœuvres, nous avons utilisé la PWM pour contrôler la vitesse de moteur des roues motrices, suivant la distance des obstacles :

Marche avant avec la vitesse maximale (100%), si la distance de l'obstacle supérieur à 2 mètres, dans ce cas 100%------- > 1023 (duty cycle).

Marche avant avec la vitesse moyenne (50%), si la distance de l'obstacle est entre 0.5 mètre et 2 mètres, dans ce cas 50%------ > 512 (duty cycle).

Pour la marche arrière, nous avons préféré qu'il aille la faire avec la vitesse moyenne de (50%) en arrière.

V.3.4.5 Les fonctions « tourne_à_droite_90° » et « tourne_à_gauche_90° »

Comme le robot est équipé de 4 roues, donc, pour qu'il puisse aller à droite ou à gauche, il doit actionner ses deux moteurs en même temps, dont le sens est dicté par la direction qu'il va prendre.

Pour tourner à droite, il convient d'envoyer le code qu'il faut sur le PORT C, et par l'expérience on peut avoir le temps nécessaire pour faire 90°. Et c'est la même chose pour tourner à gauche.

Dans notre cas, pour que le chariot tourne à droite de 90 degrés, il doit d'abord tourner à gauche en arrière pendant un moment bien spécifié, ensuite il tourne à droite pendant un temps jusqu'à avoir l'angle de 90°.

Par contre, pour qu'il tourne à gauche de 90 degrés, il doit tourner à droite en arrière pendant un moment, puis il tourne à gauche pendant un temps jusqu'à avoir l'angle de 90°.

V.3.4.5.1 La fonction « tourne_à_droite_90° »

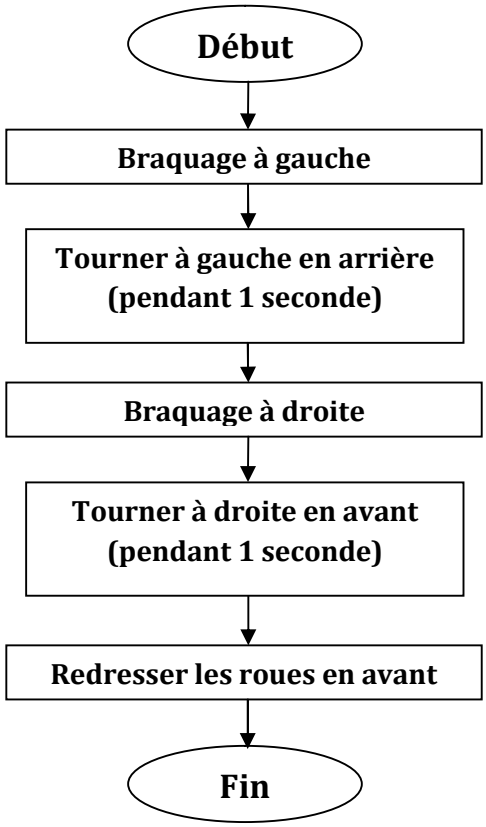


Figure V.21 :

Organigramme de la fonction « tourne_à_droite_90° »

V.3.4.5.2 La fonction « tourne_à_gauche_90° »

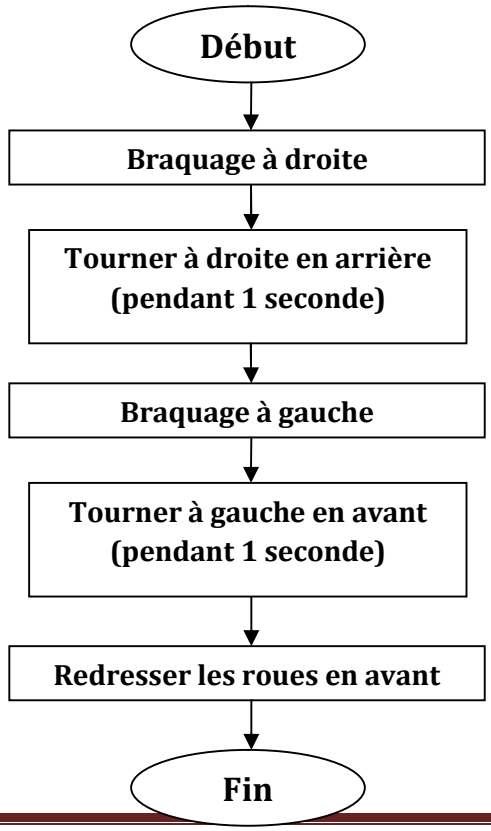


Figure V.22 :

Organigramme de la fonction « tourne_à_gauche_90° »

V.3.5 Organigramme de programme principal de la tâche (2)

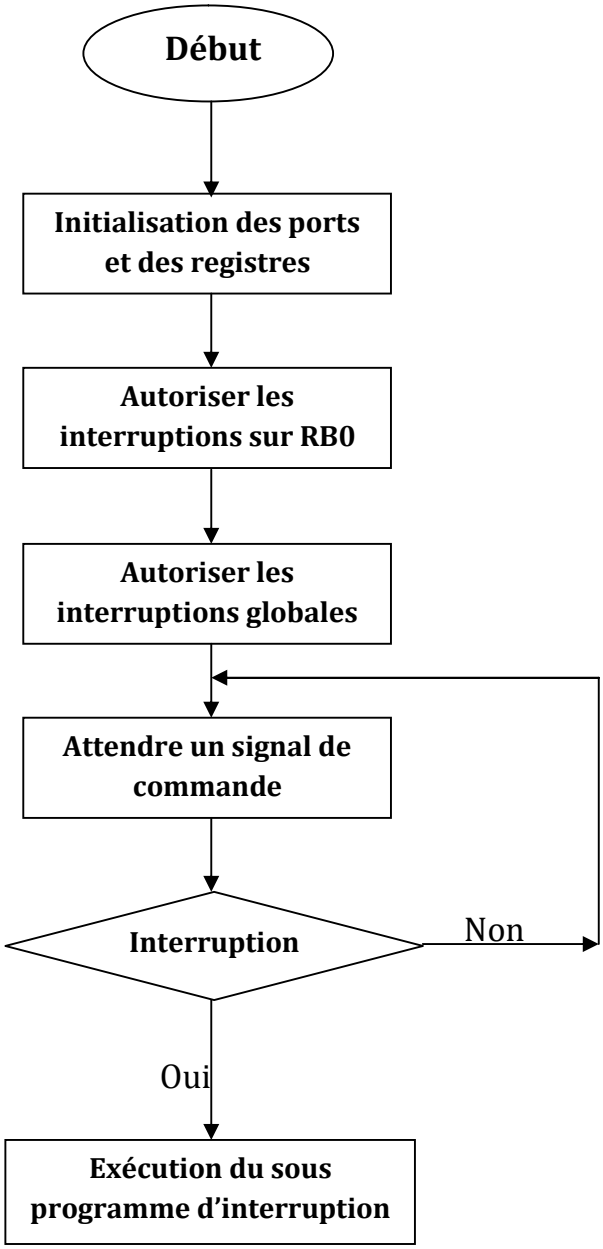
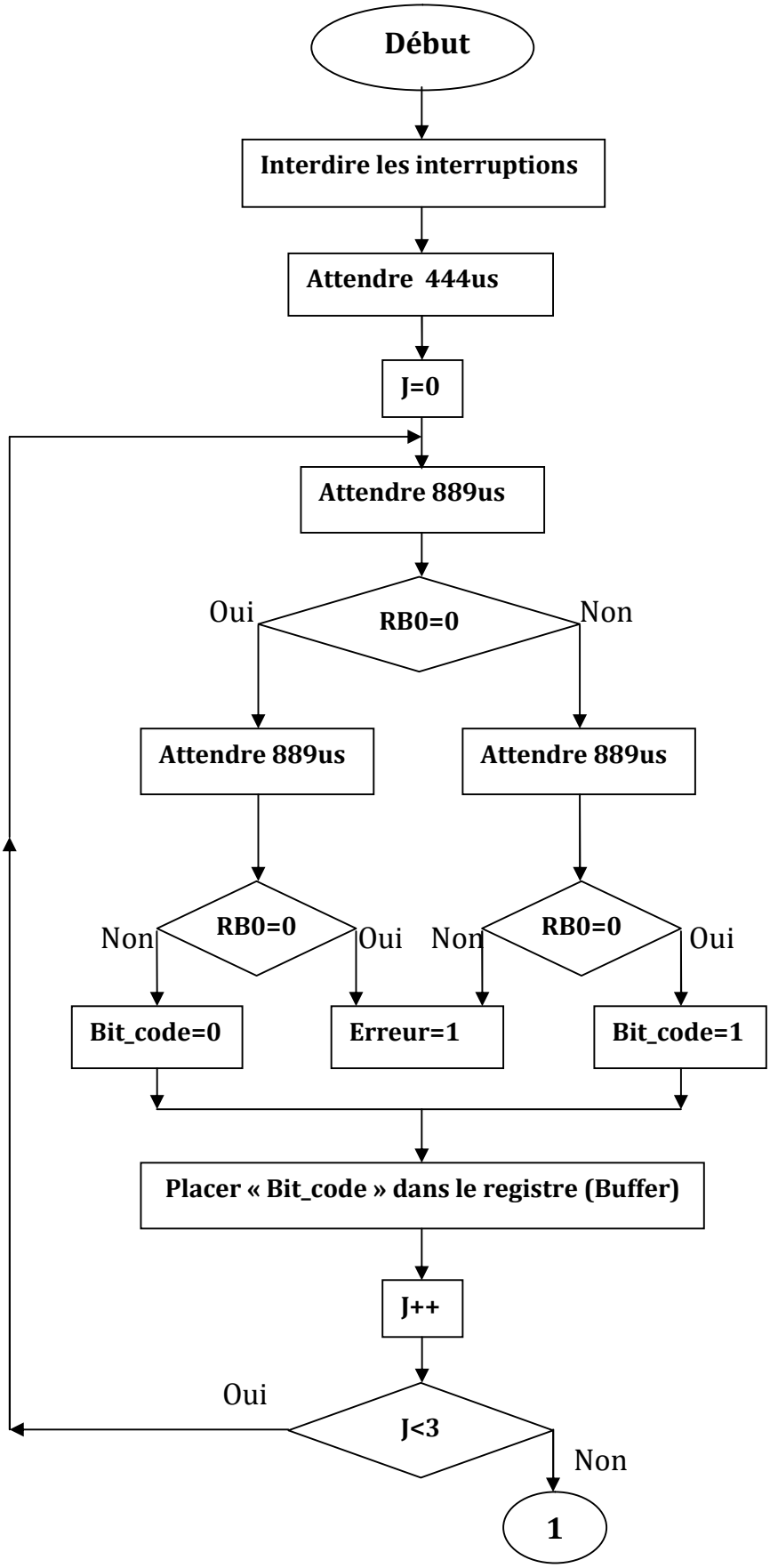


Figure V.23 : Organigramme de programme principal de la tâche 2

<Téléguidage par télécommande IR >

V.3.5.1 Organigramme de sous programme d'interruption de la tâche (2)



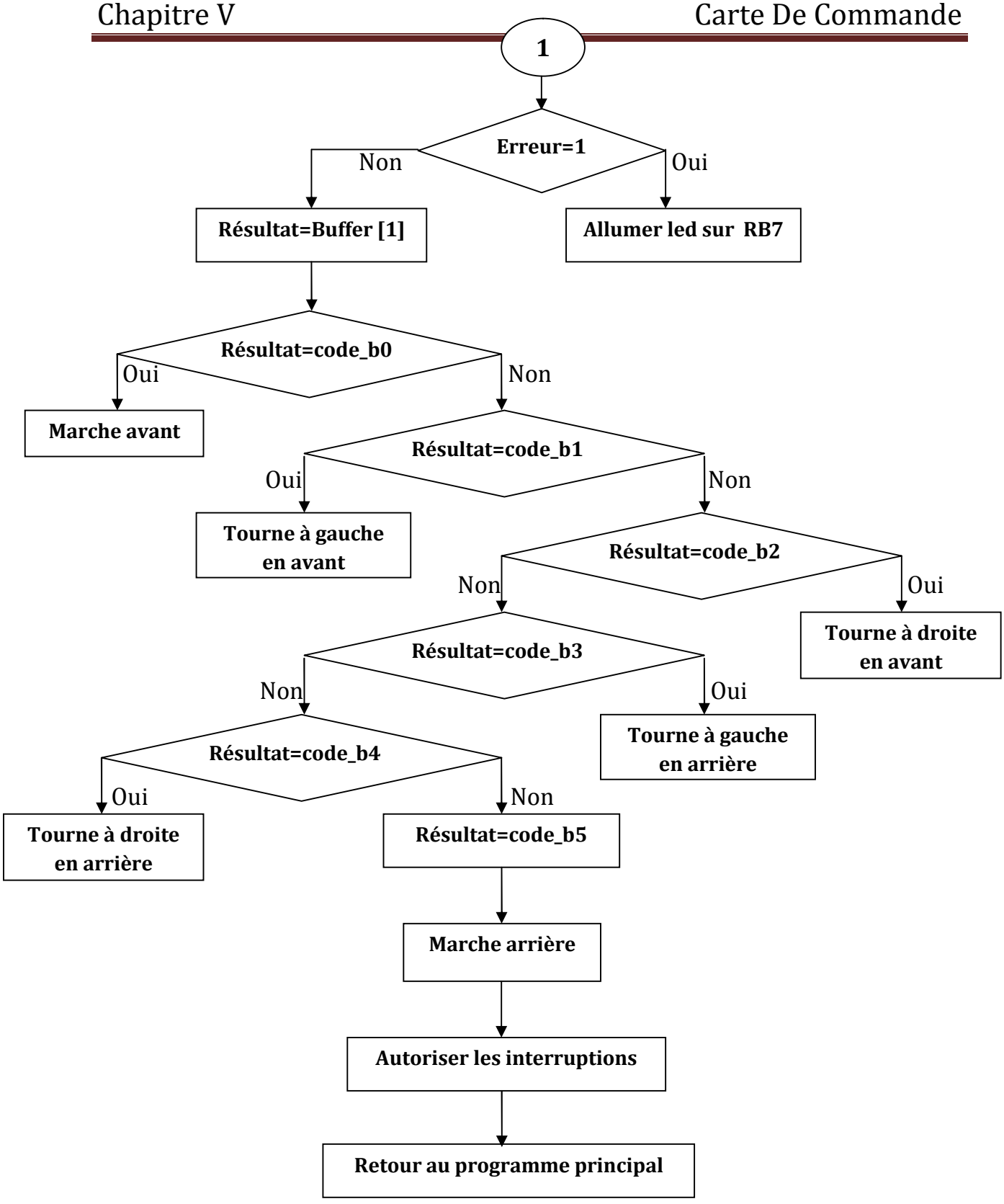
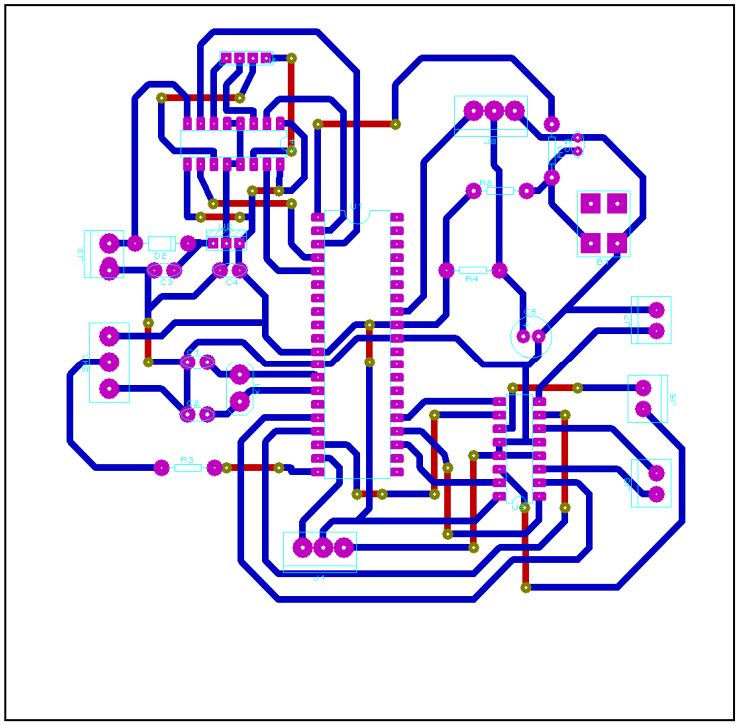


Figure V.24 : Organigramme de sous programme d'interruption de la tâche 2

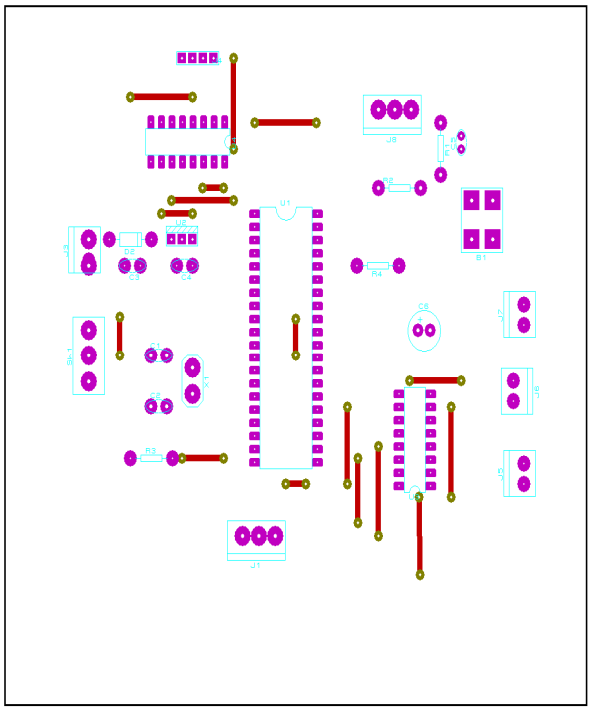
V.3.6 Réalisation pratique de la carte de commande du robot :

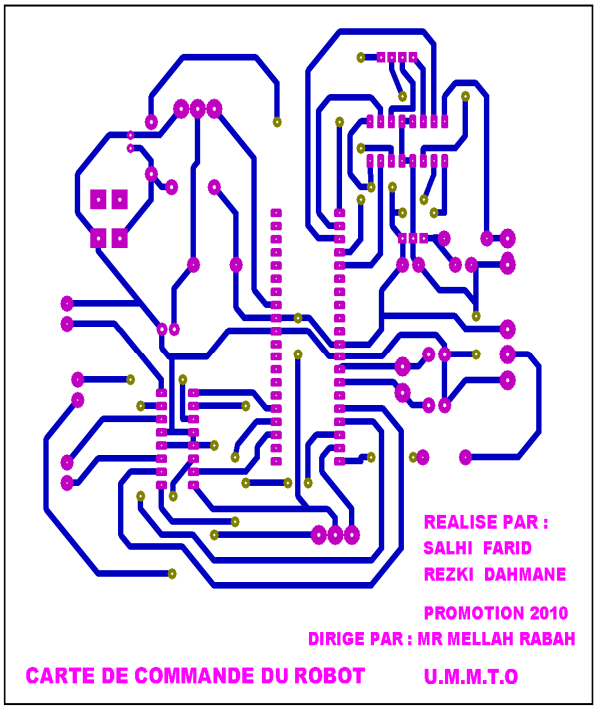
V.3.6.1 Le typon de la carte de commande :



V.3.6.2 Le circuit imprimé de la carte de commande

- Coté composants : Coté cuivre :





V.4 Conclusion

Ce chapitre nous a permet de présenter en détails les différents blocs constituant la carte à microcontrôleur que nous avons conçu pour la commande du robot mobile, ainsi, le programme de gestion sous forme des organigrammes qui décrivent la stratégie que suit le robot pour réaliser ses tâches.

CONCLUSION GÉNÉRALE

Conclusion générale

L'objectif de notre travail est la réalisation d'un robot mobile autonome qui doit explorer son environnement immédiat sans collision avec les obstacles conformément au cahier de charge dont son ensemble, à savoir commande à distance d'un chariot mobile par un microcontrôleur PIC en tenant compte de l'évitement d'obstacles.

Ce travail nous a permis de traiter des problèmes d'ordre pratique et de vérifier des connaissances théoriques acquises toute le long de notre formation.

Durant ce travail nous avons pu acquérir une certaine expérience compte à la conception et à la réalisation des différentes parties du robot.

D'un point de vu technique, malgré des difficultés certes pour réaliser l'intégration de tous les modules, et en particulier de coordonner informatique et matériel, nous avons atteint notre objectif.

Les performances de notre robot sont biens loin des performances qu'on pourra trouver sur les robots commercialisés ou ceux présentés dans des compétions.

En revanche, l'utilisation du langage C nous a permet de gagner en temps et en simplicité du programme.

Le travail qu'on a réalisé peut servir de base solide pour d'éventuels travaux qui auront comme objet la conception de carte de commande à base des microcontrôleurs de robot réalisant des tâches plus complexes.

Grâce au travail continu, on a peut atteindre notre but et satisfaire le cahier de charge, mais cela ne veut pas dire qu'il est complet. Ainsi, nous proposons que le robot réalisé soit la base de toute une série d'améliorations que nous n'avons pas eu la chance de les faire par manque de temps et de matériel.

ANNEXE

Nomenclature des composants

Carte de la télécommande de pilotage :

- **Resistances :**
  R1=100 Ω.
  R8=470 Ω.
  R2=R3=R4=R5=R6=R7=10 k Ω.
- **Capacités**
  C1=C2=15pF.
  C3=C4=4.7µF.
- **Semi-conducteurs :**
  D3 : Diode
  D1, D2 : Diodes infrarouges.
  Transistor 2N2222
- **Circuits intégrés :**
  PIC 16F84A.
  Régulateur de tension LM 7805.
- **Divers :**
  Quartz de 4 MHz.
  Batterie 9V.
  6 boutons poussoirs.

Carte de commande du robot :

- **Resistances :**
  R2=1kΩ.
  R1=R2=R4=10kΩ.
- **Capacités :**
  C3=C4=C6=C10=C9 =4.7µF.
  C5=1nF.
  C1=C2=15pF.
- **Semi-conducteurs :**
  D2, D3 Diodes.
- **Circuits intégrés :**
  PIC 16F877A.
  02 circuits L293D.
  Régulateur de tension LM 7805.
  Régulateur de tension LM 7809.
- **Divers :**
  Quartz de 4 MHz
  récepteur infrarouge TSOP 1738.
  01 bouton poussoir et un interrupteur (SW).
  Module (émetteur/récepteur) : capteur à ultrason PING))) (Parallax).

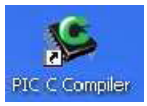
Environnement de développement et simulation du programme

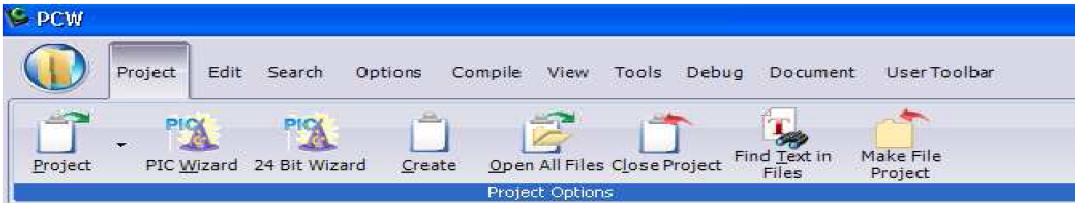
Développement

1. Choix du compilateur :

Nous avons choisi le compilateur C de CCS, dans sa version PCWH, qui est tout à la fois la plus complète et la plus performante, puisqu'elle supporte toutes les familles des Pics.

2. Le compilateur PIC C Compiler de CCS :

Une fois qu'on a installé ce programme « PIC C Compiler » sur notre PC, on démarre l'application par le raccourcie qui se trouve sur le bureau 



2.1. Création d'un nouveau projet :

Utilisation de project wizard : Menu « Project / New / PIC Wizard », ensuite sélectionner un nom de projet (fichier .PJT, il est vivement conseillé de créer un nouveau répertoire).

2.2. Configuration des propriétés du PIC

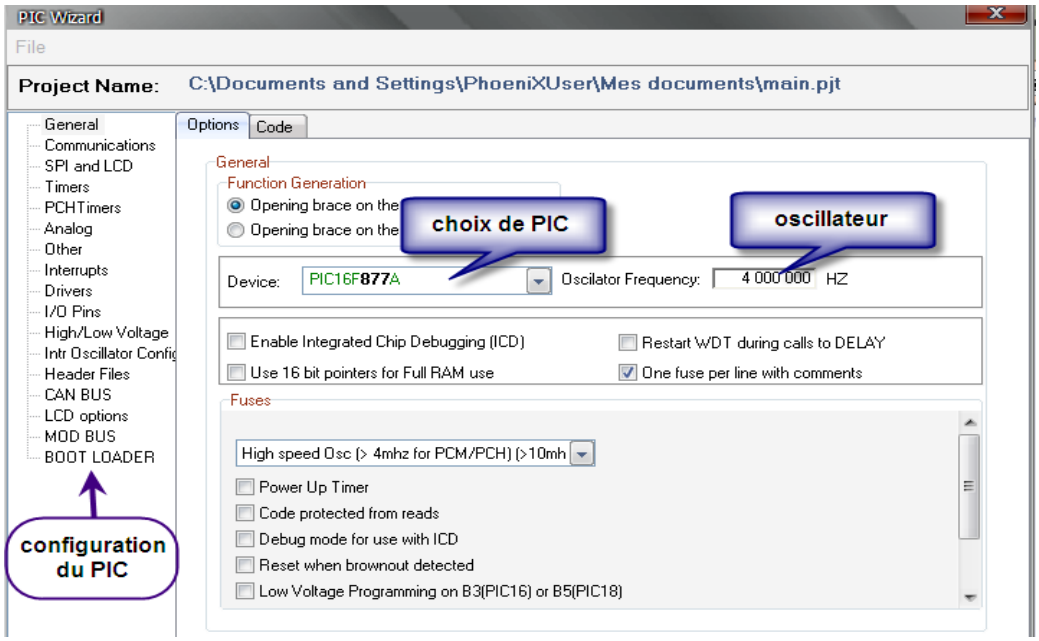


Figure 1: configuration du PIC.

Une boite à onglet apparaît, suivant les options choisies, des lignes de code seront générées automatiquement, entre autres :

- type de microcontrôleur : (pour nous, à priori : pic 16F877A)
- type d'oscillateur : crystal oscillateur, fréquence : 4 000 000 Hz.
- Communication : si utilisé, RS232#1, + paramètres corrects dans notre cas non utilisé.

Alors, cette boite nous permet de configurer notre PIC comme on veut, à savoir les: I/O, timers, interrupts, analog……..etc.

Si on a oublié quelque chose, il sera toujours temps de l'ajouter après.

2.3. Compilation du paragramme

Une fois qu'on a terminé d'écrire notre programme, on passe à la compilation.

La compilation du programme est très importante, afin d'avoir un fichier « .Hexa » exécutable par le microcontrôleur PIC.

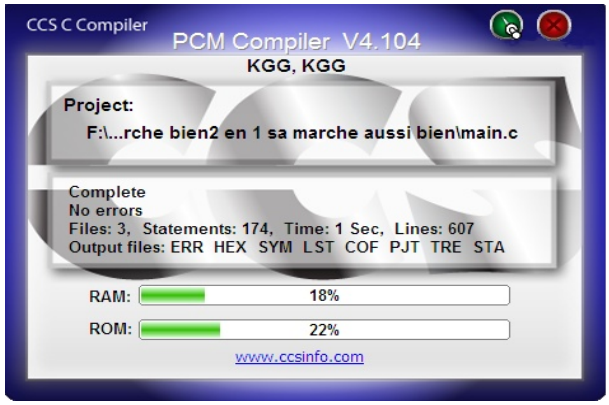


Figure 2: compilation du programme

2.4. Chargement de fichier .HEX dans le PIC :

Apres avoir écrit notre code, le compiler, au résultat de cette compilation, nous obtenons un fichier .HEX, prêt à être charger dans le PIC, grâce à un autre logiciel spécifique : « Win PIC 800 » et un programmateur de PIC de type : « JDM Programmer ».

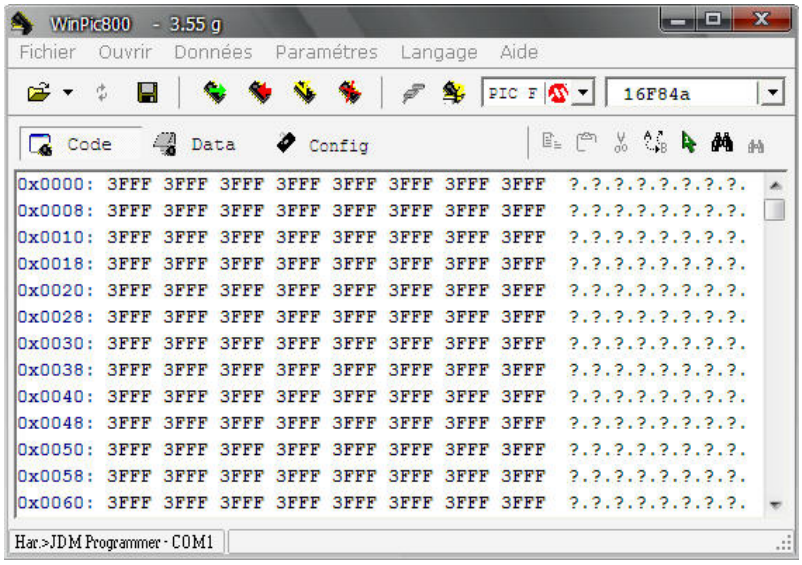


Figure 3 : Win PIC 800

(Chargement de fichier .HEX dans le PIC)

3. Présentation de logiciel PROTEUS :

PROTEUS est un ensemble de deux modules logiciels de (conception assistée par ordinateur) **CAO :**

- **ISIS :** conçu pour la réalisation et l'impression de schémas théoriques électroniques.
- **ARES :** destiné à concevoir les films ou typons permettant la gravure des circuits-imprimés, en simple, double-face ou multicouches, pour des composants standards**.**

PROTEUS est un système modulaire permettant d'une façon intégrée de:

- Saisir des schémas avec ISIS (Intelligent Schematic Input System),
- Simuler le fonctionnement du schéma avec LISA (Labcenter Simulation Integrated Architecture).
- Concevoir le circuit imprimé avec ARES (Advanced Routing and Editing Software).

Dans notre projet, on a utilisé le logiciel **PROTEUS & ARES** comme outil de CAO (conception assistée par ordinateur), qui nous a permet de saisie les schémas, les simuler et aussi de créer les circuits imprimés (PCB) à partir de ces schémas.

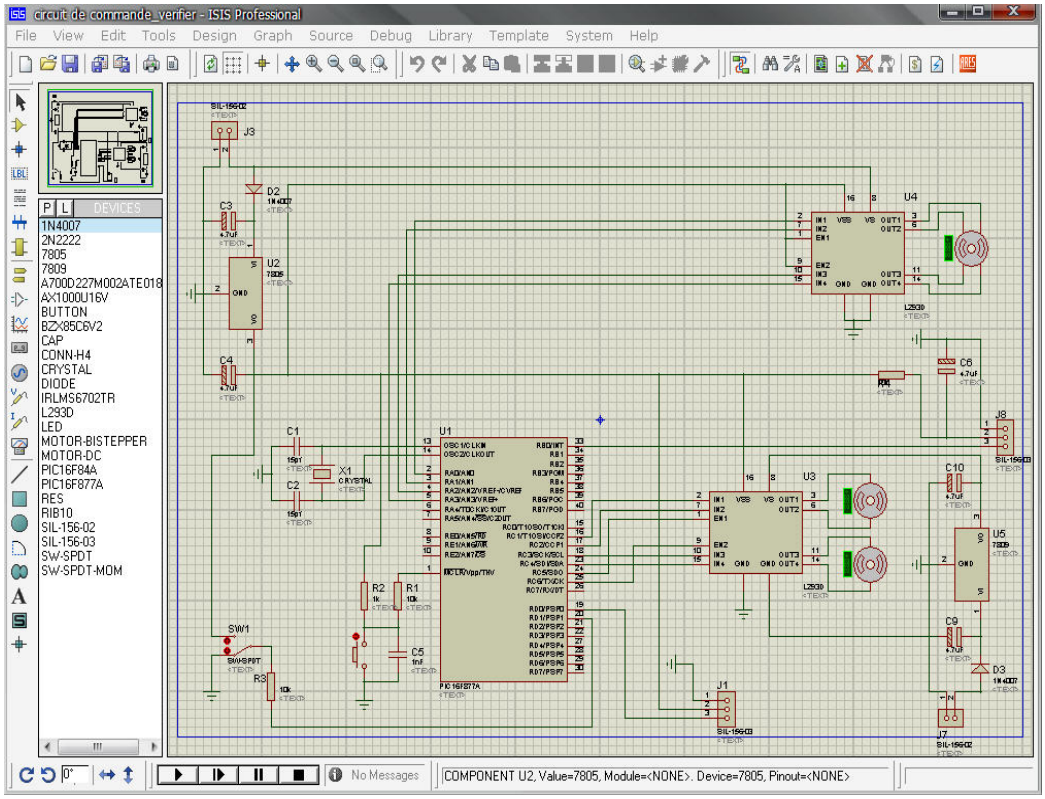


Figure 4: Le schéma de la carte de commande sous PROTEUS

Programme de transmission de la télécommande

```
//////////////Déclaration des variables//////////////////////////
int  i,j,k;
////////////////Définition des codes//////////////////////////////
byte const v1[3]={0,0,1};
byte const v2[3]={0,1,0};
byte const v3[3]={0,1,1};
byte const v4[3]={1,0,0};
byte const v5[3]={1,0,1};
byte const v6[3]={1,1,1};
////////////////Déclaration des fonctions//////////////////////////
////////////////Fonction "code_zéro"//////////////////////////////
void code_zero(void)
{
      for (i=1;i<=34;++i)
                {
                 output_high(pin_a0);
                 delay_us(6);
                   output_low(pin_a0);
                 delay_us(6);
                }
   output_low(pin_a0);
    delay_us(889);
}
///////////////fonction "code_un"////////////////////////////////
void code_un(void)
{
   output_low(pin_a0);
```

```
   delay_us(889);
      for (i=1;i<=34;++i)
                {
                 output_high(pin_a0);
                 delay_us(6);
                 output_low(pin_a0);
                 delay_us(6);
                 }
}
//////////////////Programme principal//////////////////////
void main()
{
  do
    {
//////////////////////////1ière  touche/////////////////////////
          if(input(pin_b0)==1)
           {
            code_zero() ;                    //////  c'est un bit de synchronisation
           for (j=0;j<3;++j)
               {
                if(v1[j]==1)
                  {
                    code_un();
                  }
                 else
                 {
                   code_zero();
                  }
               }
```

```
        output_low(pin_a0);
        delay_ms(89);
        }
//////////////////////////2iéme touche//////////////////////
        if(input(pin_b1)==1)
         {
           code_zero() ;                  //////  c'est un bit de synchronisation
           for (j=0;j<3;++j)
              {
               if(v2[j]==1)
                 {
                  code_un();
                 }
                else
                {
                 code_zero();
                }
              }
        output_low(pin_a0);
        delay_ms(89);
    }
//////////////////////////3iéme touche//////////////////////
        if(input(pin_b2)==1)
         {
           code_zero() ;                //////  c'est un bit de synchronisation
           for (j=0;j<3;++j)
              {
                if(v3[j]==1)
                  {
```

```
                code_un();
                }
               else
                {
                  code_zero();
                }
             }
        output_low(pin_a0);
        delay_ms(89);
    }
//////////////////////////4iéme touche//////////////////////
        if(input(pin_b3)==1)
        {
          code_zero() ;                    //////  c'est un bit de synchronisation
           for (j=0;j<3;++j)
             {
               if(v4[j]==1)
                {
                code_un();
                }
               else
                {
                 code_zero();
                }
             }
        output_low(pin_a0);
        delay_ms(89);
    }
```

```
//////////////////////////5iéme touche//////////////////////
        if(input(pin_b4)==1)
        {
          code_zero();              ////// c'est un bit de synchronisation
          for (j=0;j<3;++j)
             {
             if(v5[j]==1)
                 {
                  code_un();
                 }
               else
               {
                code_zero();
               }
             }
        output_low(pin_a0);
       delay_ms(89);
    }
//////////////////////////6iéme touche //////////////////////
         if(input(pin_b5)==1)
         {
           code_zero();              ////// c'est un bit de synchronisation
           for (j=0;j<3;++j)
              {
              if(v6[j]==1)
                 {
                  code_un();
                 }
               else
```

```
                    {
                     code_zero();
                    }
                 }
          output_low(pin_a0);
          delay_ms(89);
      }
}
while(true);    //bouclage de programme
}
```

Programme de la carte de commande du robot

```
//////////////////////Déclaration des variables//////////////////////////
int32 dist,dist_droite,dist_gauche,dist_devant,temps;
int16 cmp=0;
int i,j,erreur;
int8 resultat;
int1 bit_code;
byte buffer[1];
//////////////////////Définition des codes//////////////////////
#define code_b0 0x01
#define code_b1 0x02
#define code_b2 0x03
#define code_b3 0x04
#define code_b4 0x05
#define code_b5 0x07
#define REC_IR pin_b0
//////////////////////les sous programmes d'interruption//////////////
//////////////////////interruption RB0//////////////////////////
#int_EXT
void  EXT_isr(void)
{
  disable_interrupts(INT_EXT);
  delay_us(444);
  for (i=0;i<3;++i)
     {
      delay_us(889);
      if(input(REC_IR)==0&&erreur==0)
        {
```

```
        delay_us(889);
        if(input(REC_IR)!=1)
          {
           erreur=1;
          }
        bit_code=0;
        }
      else if (input(REC_IR)==1&&erreur==0)
        {
         delay_us(889);
         if(input(REC_IR)!=0)
           {
           erreur=1;
            }
         bit_code=1;
         }
         shift_left(buffer,1,bit_code);
      }
  if(erreur==1)
       {
       output_high(pin_b7);
       }
       else
       {
       resultat=buffer[0];
      switch(resultat)
       {
      case code_b0:
          output_c(0x24);
```

```
    delay_ms(500);
     output_c(0x60);
  break;
case code_b5:
     output_c(0x22);
     delay_ms(500);
     output_c(0x60);
  break;
case code_b2:
     output_c(0x6c);
     delay_ms(500);
     output_c(0x60);
  break;
case code_b1:
     output_c(0x74);
     delay_ms(500);
     output_c(0x60);
  break;
case code_b4:
     output_c(0x6a);
     delay_ms(500);
     output_c(0x60);
  break;
case code_b3:
     output_c(0x72);
     delay_ms(500);
     output_c(0x60);
  break;
  }
```

```
      }
}
//////////////////////////interruption  TIMER 1//////////////////////////
#int_TIMER1
void  TIMER1_isr(void)
    {
      cmp++;
    }
//////////////////////////////Déclaration des fonctions//////////////////
//////////Fonction calcul de distance avec le capteur à ultrason PING))/////////
void distance(void)
{
 setup_timer_1(T1_INTERNAL);
 enable_interrupts(int_timer1);
 enable_interrupts(global);
 output_low(pin_d0);
 delay_ms(5);
 output_high(pin_d0);
 delay_us(5);
 output_low(pin_d0);
 delay_us(700);
 while(input(pin_d0)==0){};
 set_timer1(0);
 cmp = 0;
 while(input(pin_d0)==1)  {  };
disable_interrupts(global);
 temps = get_timer1();
 temps = ((temps +(cmp*65535.00))-20);
dist =0.034 * (temps/2);          }
```

///////////////les fonctions de commande du moteur pas à pas///////////////

//////////////////le moteur pas à pas tourne à droite de 90°//////////////

```
    void MPP_tourne_droite(void)
{
   const char d1[4]={0x06,0x0a,0x09,0x05};   // séquence de commande
        for (j=0;j<=2;++j)
            {
            for (i=0;i<=3;++i)
               {
                output_a(d1[i]);
                delay_ms(50);
              }
            }
}
```

//////////////////le moteur pas à pas tourne à gauche de 180°///////////////

```
void MPP_tourne_gauche(void)
{
const char g[4]={0x09,0x0a,0x06,0x05};     //séquence de commande
    for (j=0;j<=5;++j)
        {
        for (i=0;i<=3;++i)
            {
             output_a(g[i]);
             delay_ms(50);
            }
        }
}
```

//////le moteur pas à pas retourne à droite de 90°(à sa position initiale)///////

```
 void MPP_tourne_devant(void)
{
const char d2[4]={0x06,0x0a,0x09,0x05};   //séquence de commande
    for (j=0;j<=2;++j)
        {
         for (i=0;i<=3;++i)
             {
                output_a(d2[i]);
                delay_ms(50);
              }
         }
}
```

////////////////la fonction tourne_à_droite_90° du chariot ////////////////

```
void   tourne_à_droite_90°(void)
{
            output_c(0x70);
            delay_ms(1000);
            output_c(0x72);
            delay_ms(1000);
            output_c(0x74);
            delay_ms(100);
            output_c(0x20);
            delay_ms(500);
            output_c(0x48);
            delay_ms(1000);
            output_c(0x6c);
            delay_ms(1000);
            output_c(0x20);          }
```

```
////////////////la fonction  tourne_à_gauche_90° du chariot ////////////////
void   tourne_à_gauche_90°(void)
{
            output_c(0x48);
            delay_ms(1000);
            output_c(0x6a);
            delay_ms(1000);
            output_c(0x6c);
            delay_ms(100);
            output_c(0x20);
            delay_ms(500);
            output_c(0x50);
            delay_ms(1000);
            output_c(0x74);
            delay_ms(1000);
            output_c(0x20);
}

////////////////////////Programme principal//////////////////////////
void main()
{
setup_timer_2(T2_DIV_BY_16,243,1);
debut:
///////////////// Tâche  01 "évitement d'obstacle"//////////////////////////
  if(input(pin_d1)==0)
    {
     distance();
    dist_devant=dist;
```

```
 if(dist_devant>=50)
    {
      setup_ccp1(CCP_pwm);
      setup_ccp2(CCP_off);
      output_high(pin_c5);
      output_low(pin_c6);
  if(dist_devant>=50&&dist_devant<=200)
  {
   output_low(pin_c1);
   set_pwm1_duty(512);
   goto debut;
  }
  if(dist_devant>200)
  {
   output_low(pin_c1);
   set_pwm1_duty(1023);
   goto debut;
  }
   }
 else
  {
loop:  setup_ccp2(CCP_off);
       setup_ccp1(CCP_off);
       output_c(0x22);
       delay_ms(300);          // système de freinage en avant
       output_c(0x26);
       MPP_tourne_droite();
       delay_ms(500);
```

```
            distance();
            dist_droite=dist;
            MPP_tourne_gauche();
            delay_ms(500);
           distance();
           dist_gauche=dist;
           MPP_tourne_devant() ;
   if(dist_droite>=dist_gauche&&dist_droite>50)
    {
             tourne_à_droite_90°() ;
             goto debut;
    }
    else if(dist_gauche>dist_droite&&dist_gauche>50)
     {
             tourne_à_gauche_90°() ;
             goto debut;
     }
      setup_ccp1(CCP_off);
      setup_ccp2(CCP_pwm);
      output_low(pin_c2);
      set_pwm2_duty(512);
      delay_ms(1000);
      goto loop;
  }
}
```

////////////////////// Tâche 02 "pilotage par télécommande"///////////////

```
if(input(pin_d1)==1)
  {
  buffer[0]=0;
  erreur=0;
  i=0;
  enable_interrupts(INT_EXT);
  enable_interrupts(GLOBAL);
  }
goto debut;   // bouclage du programme
}
```

Présentation de circuit intégré L293D

1) Présentation de L293D

Ce circuit intégré est destiné à la commande des moteurs de faible puissance .il permet, à partir d'un circuit numérique, la commande soit d'un moteur pas à pas bipolaire, soit deux moteurs à courant continu dans les deux sens.

2) Structure interne

Comme illustré dans la figure suivante, le L293D est constitué de deux étage identique correspondant chacun à un pont en H. chaque étage comporte deux broches numériques d'entrée, une broche numérique d'activation (ENABLE) ainsi que deux broches de sortie « analogiques »pour la commande des moteurs.

Les broches OUT1 et OUT2 (ou 3 et 4) sont les sorties analogiques de l'étage : elles fournissent la tension Vs selon la polarité voulue (en fonction de l'état des broches d'entrée (INPUT) de commande)

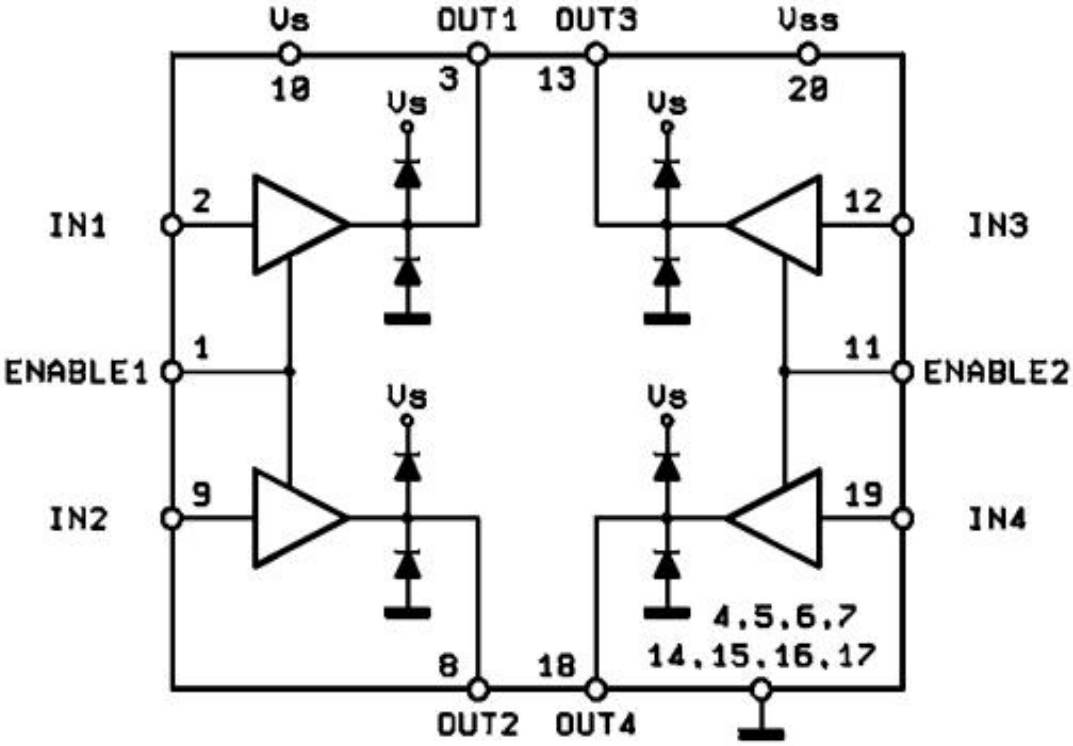


Figure 5: Structure interne du L293D>

3) Fonctionnement

Le L293D est constitué de deux « pont en H ». Un pont en H est un circuit électronique qui permet, à l'aide d'entrée numérique, d'inverser la polarité aux bornes d'un moteur ou d'une bobine.

Pour le cas de commande d'un moteur à courant continu, les sorties sont couplées deux par deux (les sorties 1 et 2 fonctionnent ensemble de même pour les sorties 3 et 4).

Si on applique respectivement, les niveaux 0 et 1 sur input1 et input2, la sortie sera 0V sur output1 et +Vs sur output2. Inversement, si on applique respectivement, les niveaux 1 et 0 sur input1 et input2, la sortie sera +Vs sur output1 et 0V sur output2 (la même chose pour le deuxième étage avec les inputs et outputs 3 et 4).

4) Brochage

Ce composant se présente sous forme de boitier DIL16 classique. On peut classer les broches en différents groupes comme suit :

- **Broches d'alimentation :**

  - Vss : broche d'alimentation logique.
  - Vs : broche d'alimentation analogique de puissance pouvant atteindre 36 V.
  - GND : sont les broches de masses communes à l'alimentation logique et analogique.

➢ **Broches communes à chaque étage :**

- Broche d'entrée numérique « ENABLE » : elle active l'étage sur le niveau haut (+5V).
- Broches input 1 et 2 (ou 3 et 4) : sont les entrées numériques de l'étage. Elles servent à commander la polarité des broches de sorties de l'étage.

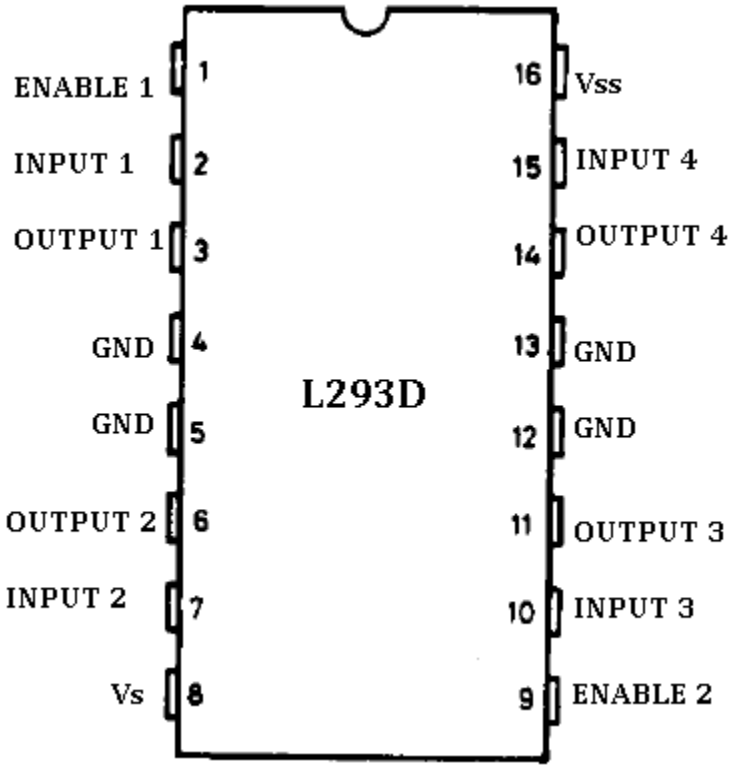


Figure : Brochage du L293D

5) Principe de montage du L293D avec deux moteurs à courant continu

Comme le montre la figure suivante, le principe du montage d'un L293D avec deux moteurs à courant continu :

- On connecte Vss au +5V.
- On connecte Vs à l'alimentation des moteurs.
- On connecte les broches ENABLE au +5v pour valider toujours les étages.
- On connecte les broches INPUT sur les broches I/O (ou E/S) de µC.
- On connecte les broches OUTPUT sur les broches d'alimentation des deux moteurs.

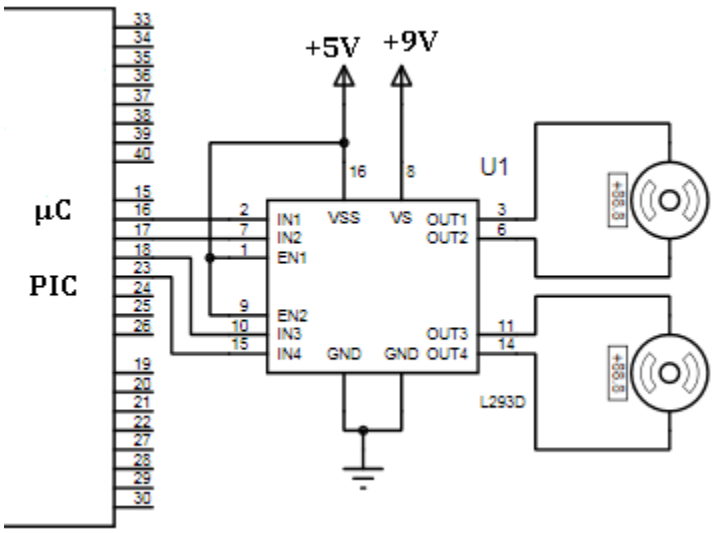


Figure : Principe de montage du L293D avec deux moteurs à CC

6) Caractéristiques électriques

- Tension d'alimentation du circuit moteur peut atteindre les 36 V.
- Intensité supportée par chaque étage : 600mA, en continu jusqu'à 1.2A.
- Entrée numérique compatibles TTL et CMOS.
- Les diodes de protection (des moteurs) sont intégrées dans le boîtier ce qui simplifie énormément le montage.

BIBLIOGRAPHIE

[1] CHRISTIAN TAVERNIER

« Programmation en C des PIC »

Edition DUNOD 2005.

[2] CHRISTIAN TAVERNIER

« Les microcontrôleurs PIC,

Description et mise en œuvre»

Edition DUNOD 2000.

[3] BIGONOFF

« La programmation des PICS »

La première et la deuxième partie.

[4] ALAIN PRUSKI

« Robotique Générale »

Edition ellipses 1989

[5] SEDKI OMAR, TAMI RAMDANE

«Conception et réalisation d'une carte de commande

D'un robot mobile téléguidé par infrarouge»

Mémoire ingénieur d'état en automatique, encadré par : Mr MELLAH .R.

Université UMMTO Tizi-Ouzou 2009.

[6] TRIKI AHCENE

«Conception d'une carte de commande pour un robot mobile

à base du PIC 16F84A »

Mémoire ingénieur d'état en automatique, encadré par : Mr DIRAMI.

Université UMMTO Tizi-Ouzou 2004.

[7]HAROUCHE HAYET, IABBOUTENE KARIMA

« Étude et réalisation d'une commande d'un banc de perçage

à base du microcontrôleur PIC 16F876 »

Mémoire ingénieur d'état en automatique, encadré par : Mr AKROUF.

Université UMMTO Tizi-Ouzou 2007.

[8] Electronique et Loisirs Magazine

Application : « Circuit pour moteur pas à pas »

Revue d'électronique n° 09 février 2000.

[9] Internet :

www.Parallax.com

www.génération-robots.com

Résumé

La robotique est le domaine où se conjuguent plusieurs spécialités en vue de réaliser une entité pourvue de possibilités d'interagir avec son environnement, par une intervention minimale d'un operateur humain. Selon l'importance de la sollicitation en question, le robot acquiert un degré spécifique d'autonomie. Il existe différentes catégories de robots, et celle qui a fait l'objet du présent travail est la robotique mobile.

Les robots mobiles prennent actuellement une place importante dans la vie quotidienne du grand public. Notamment dans le domaine de la sécurité civile divers prototypes, chargés de pénétrer dans des endroits hostiles en cas d'incendie, de tremblement de terre etc.… De plus, d'autres applications comme la conquête spatiale fait intervenir principalement des robots mobiles dont la mission opportunité est la dernière en date. Néanmoins, toutes ces applications ne pouvaient pas être mises en pratique sans avoir associe aux robots mobiles le concept de la commande à distance.

Le contexte dans lequel s'inscrit notre travail, consiste à concevoir et réaliser un système de commande automatisé à distance d'un chariot mobile en tenant compte de l'évitement d'obstacles. La commande à distance s'effectue via une télécommande infrarouge à base de deux microcontrôleurs, dont l'un en émission et l'autre en réception. Celui de l'émission permet de générer le code approprié à une touche appuyée en utilisant le code RC5, et le transmettre par la suite au récepteur à travers une diode infrarouge. Quant à celui du récepteur a pour objectif de décoder le message reçue, par le récepteur infrarouge TSOP 1738 et d'exécuter la tâche appropriée, permettant ainsi au chariot d'effectuer l'un des mouvements à savoir, rouler en marche avant et en arrière selon l'orientation à gauche ou à droite, tout en assurant les fins de course.

Pour permettre au chariot d'éviter les obstacles, nous avons doté notre système de commande d'un capteur à ultrason qui permet de calculer la distance sans contact avec les obstacles. Ce capteur à ultrason est entrainé par un moteur pas à pas de telle manière, dés que le chariot se rapproche d'un obstacle, il s'arrête, tester la voie libre à suivre (gauche ou droite) afin de contourner l'obstacle.

Pour gérer l'ensemble des circuits de commande, nous avons utilisé une unité de traitement puissante, un microcontrôleur PIC 16F877A, qui est programmé pour atteindre le but de la réalisation (le téléguidage par télécommande et l'évitement d'obstacles).

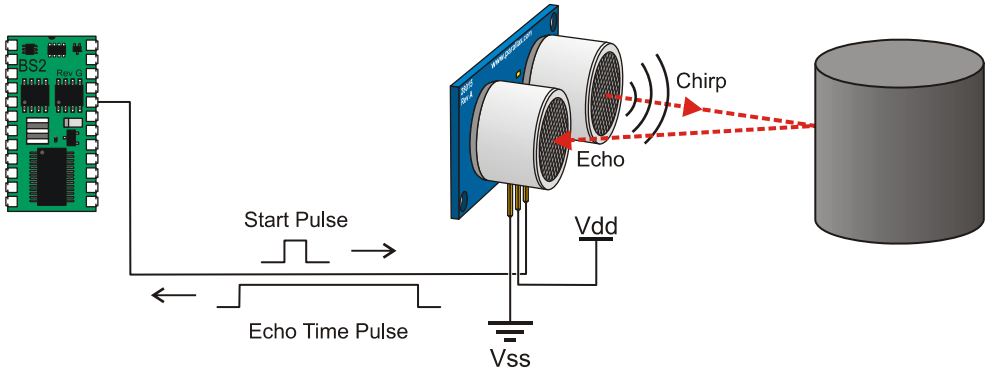
Web Site: www.parallax.com
Forums: forums.parallax.com
Sales: sales@parallax.com
Technical: support@parallax.com

Office: (916) 624-8333
Fax: (916) 624-8003
Sales: (888) 512-1024
Tech Support: (888) 997-8267

PING)))™ Ultrasonic Distance Sensor (#28015)

The Parallax PING))) ultrasonic distance sensor provides precise, non-contact distance measurements from about 2 cm (0.8 inches) to 3 meters (3.3 yards). It is very easy to connect to microcontrollers such as the BASIC Stamp®, SX or Propeller chip, requiring only one I/O pin.

The PING))) sensor works by transmitting an ultrasonic (well above human hearing range) burst and providing an output pulse that corresponds to the time required for the burst echo to return to the sensor. By measuring the echo pulse width, the distance to target can easily be calculated.



Features

- Range: 2 cm to 3 m (0.8 in to 3.3 yd)
- Burst indicator LED shows sensor activity
- Bidirectional TTL pulse interface on a single I/O pin can communicate with 5 V TTL or 3.3 V CMOS microcontrollers
- Input trigger: positive TTL pulse, 2 µs min, 5 µs typ.
- Echo pulse: positive TTL pulse, 115 µs minimum to 18.5 ms maximum.
- RoHS Compliant

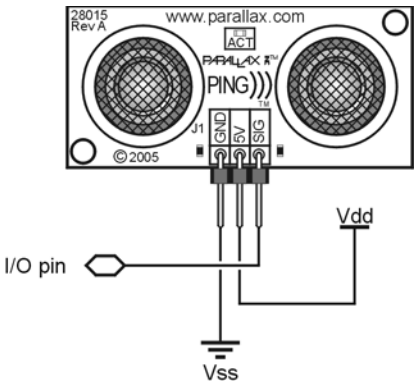
Key Specifications

- Supply voltage: +5 VDC
- Supply current: 30 mA typ; 35 mA max
- Communication: Positive TTL pulse
- Package: 3-pin SIP, 0.1" spacing (ground, power, signal)
- Operating temperature: 0 – 70° C.
- Size: 22 mm H x 46 mm W x 16 mm D (0.84 in x 1.8 in x 0.6 in)
- Weight: 9 g (0.32 oz)

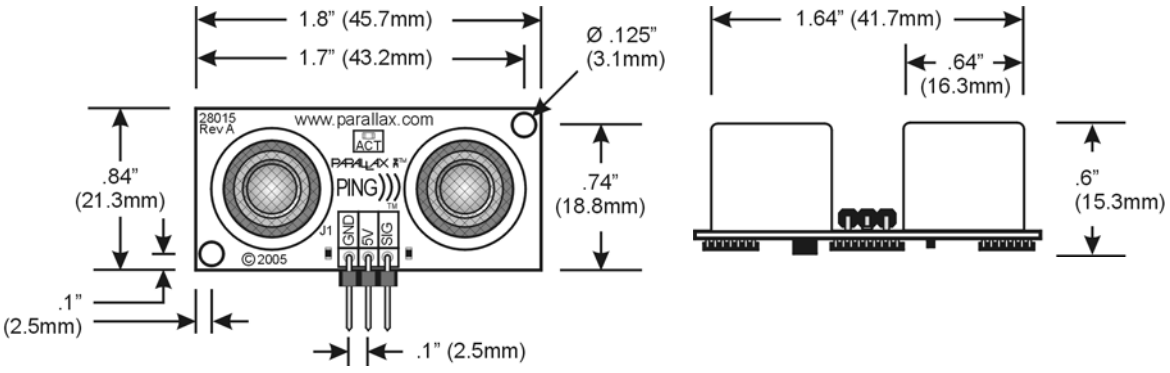
Pin Definitions

GND	Ground (Vss)
5 V	5 VDC (Vdd)
SIG	Signal (I/O pin)

The PING))) sensor has a male 3-pin header used to supply ground, power (+5 VDC) and signal. The header may be plugged into a directly into solderless breadboard, or into a standard 3-wire extension cable (Parallax part #805-000012).

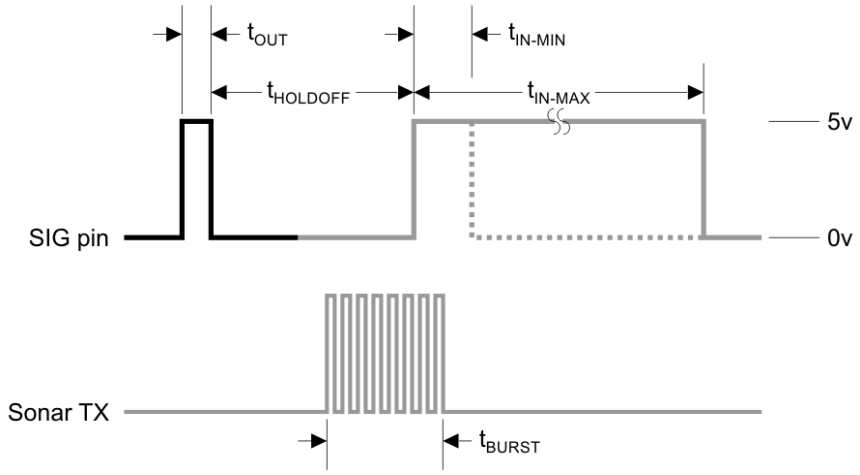


Dimensions



Communication Protocol

The PING))) sensor detects objects by emitting a short ultrasonic burst and then "listening" for the echo. Under control of a host microcontroller (trigger pulse), the sensor emits a short 40 kHz (ultrasonic) burst. This burst travels through the air, hits an object and then bounces back to the sensor. The PING))) sensor provides an output pulse to the host that will terminate when the echo is detected, hence the width of this pulse corresponds to the distance to the target.

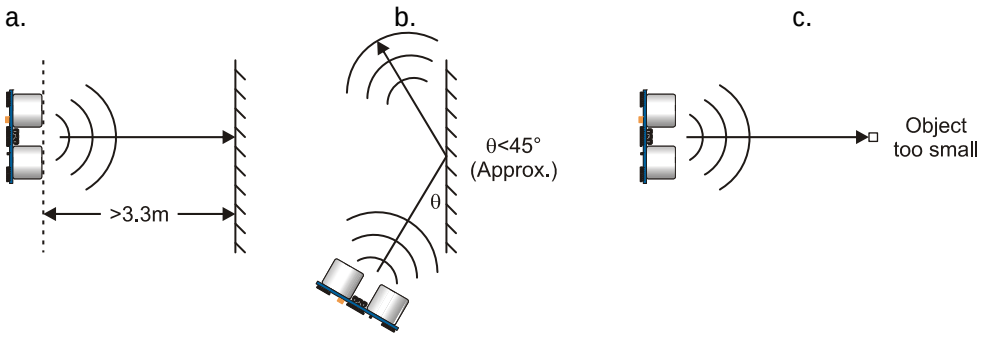


▬	Host Device	Input Trigger Pulse	t_{OUT}	2 µs (min), 5 µs typical
▬	PING))) Sensor	Echo Holdoff	$t_{HOLDOFF}$	750 µs
		Burst Frequency	t_{BURST}	200 µs @ 40 kHz
		Echo Return Pulse Minimum	t_{IN-MIN}	115 µs
		Echo Return Pulse Maximum	t_{IN-MAX}	18.5 ms
		Delay before next measurement		200 µs

Practical Considerations for Use

Object Positioning

The PING))) sensor cannot accurately measure the distance to an object that: a) is more than 3 meters away, b) that has its reflective surface at a shallow angle so that sound will not be reflected back towards the sensor, or c) is too small to reflect enough sound back to the sensor. In addition, if your PING))) sensor is mounted low on your device, you may detect sound reflecting off of the floor.



Target Object Material

In addition, objects that absorb sound or have a soft or irregular surface, such as a stuffed animal, may not reflect enough sound to be detected accurately. The PING))) sensor will detect the surface of water, however it is not rated for outdoor use or continual use in a wet environment. Condensation on its transducers may affect performance and lifespan of the device.

Air Temperature

Temperature has an effect on the speed of sound in air that is measurable by the PING))) sensor. If the temperature (°C) is known, the formula is:

$$C_{air} = 331.5 + (0.6 \times T_C)\ m/s$$

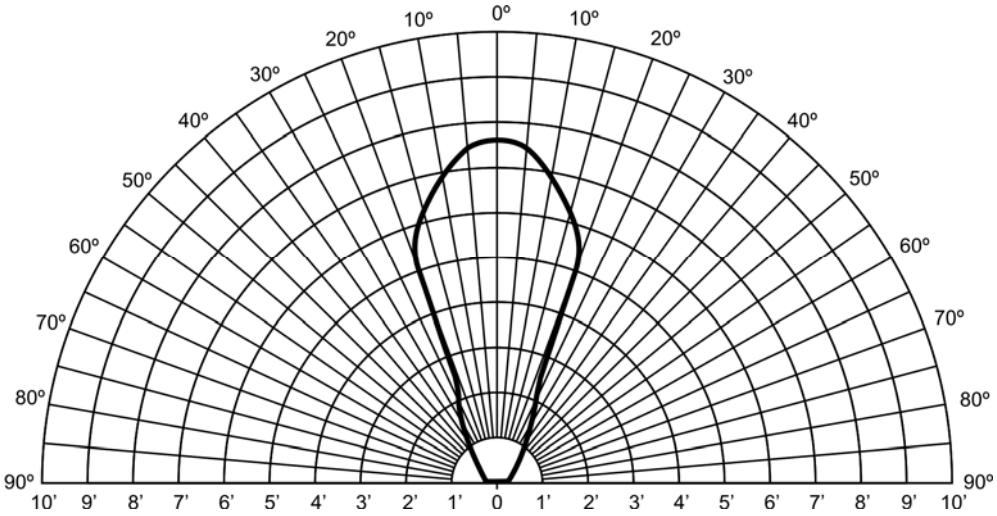
The percent error over the sensor's operating range of 0 to 70 ° C is significant, in the magnitude of 11 to 12 percent. The use of conversion constants to account for air temperature may be incorporated into your program (as is the case in the example BS2 program given in the Example Programs section below). Percent error and conversion constant calculations are introduced in Chapter 2 of *Smart Sensors and Applications*, a Stamps in Class text available for download from the 28029 product page at www.parallax.com.

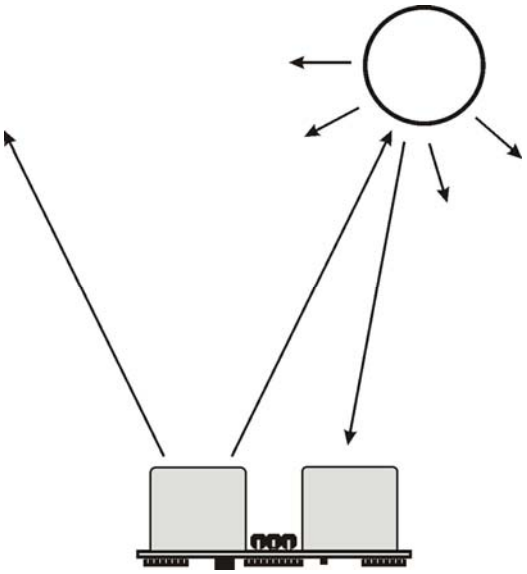
Test Data

The test data on the following pages is based on the PING))) sensor, tested in the Parallax lab, while connected to a BASIC Stamp microcontroller module. The test surface was a linoleum floor, so the sensor was elevated to minimize floor reflections in the data. All tests were conducted at room temperature, indoors, in a protected environment. The target was always centered at the same elevation as the PING))) sensor.

Test 1

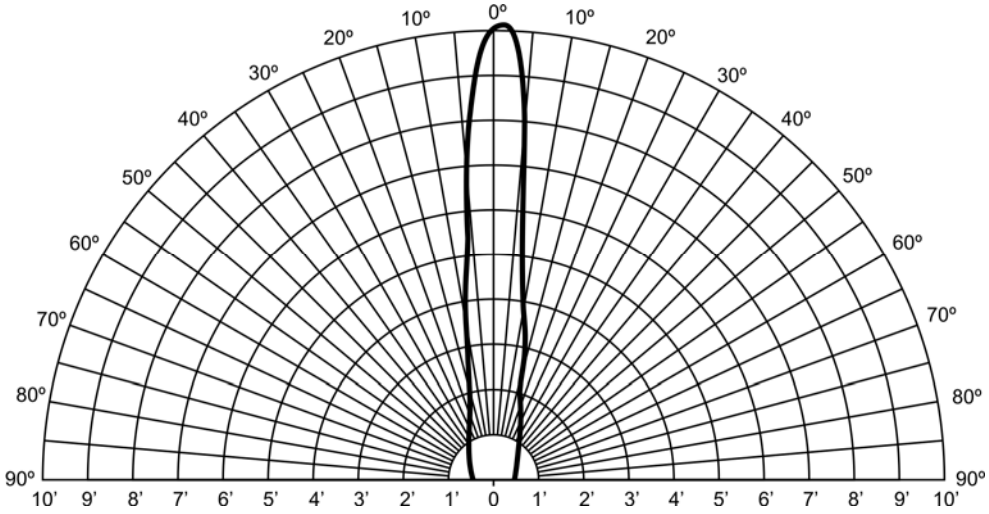
Sensor Elevation: 40 in. (101.6 cm)
Target: 3.5 in. (8.9 cm) diameter cylinder, 4 ft. (121.9 cm) tall – vertical orientation

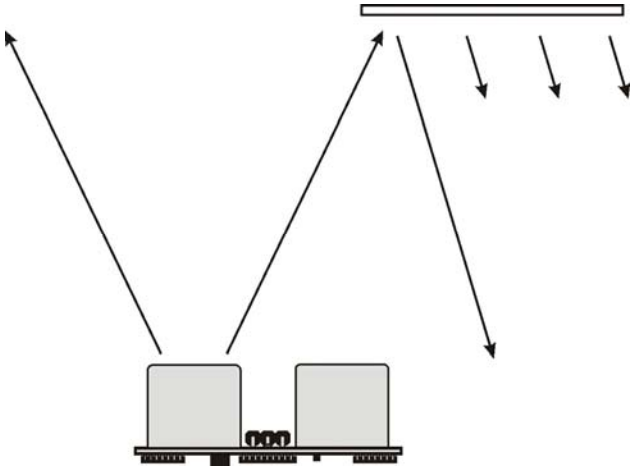




Test 2

Sensor Elevation: 40 in. (101.6 cm)
Target: 12 in. x 12 in. (30.5 cm x 30.5 cm) cardboard, mounted on 1 in. (2.5 cm) pole
Target positioned parallel to backplane of sensor

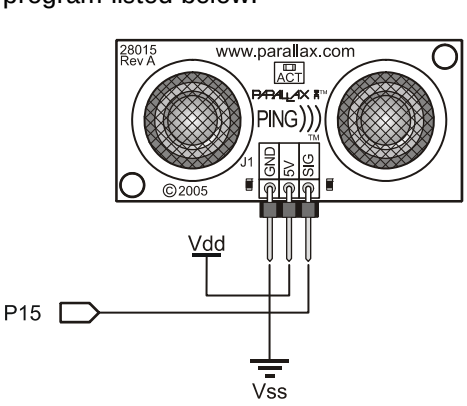


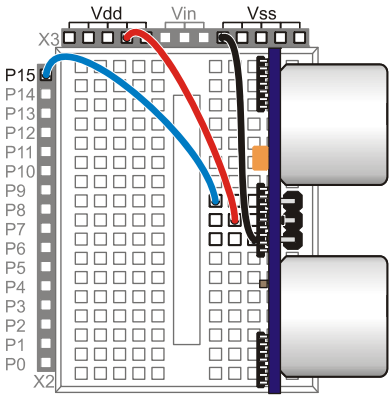


Example Programs and Applications

BASIC Stamp 2

This circuit allows you to quickly connect your PING))) sensor to a BASIC Stamp® 2 via the Board of Education® breadboard area. The PING))) module's GND pin connects to Vss, the 5 V pin connects to Vdd, and the SIG pin connects to I/O pin P15. This circuit will work with the example BASIC Stamp program listed below.

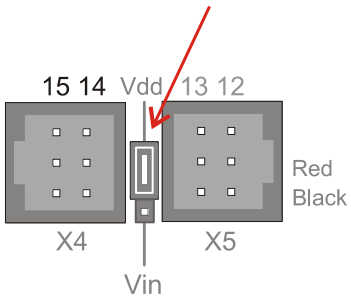




Extension Cable and Port Cautions for the Board of Education

If you are connecting your PING))) sensor to a Board of Education platform using an extension cable, follow these steps:

1. When plugging the cable onto the PING))) sensor, connect Black to GND, Red to 5 V, and White to SIG.
2. Check to see if your Board of Education servo ports have a jumper, as shown at right.
3. If your Board of Education servo ports have a jumper, set it to Vdd as shown. Then plug the cable into the port, matching the wire color to the labels next to the port.
4. If your Board of Education servo ports do not have a jumper, **do not use them with the PING))) sensor**. These ports only provide Vin, not Vdd, and this may damage your PING))) sensor. Go to the next step.
5. Connect the cable directly to the breadboard with a 3-pin header as shown above. Then, use jumper wires to connect Black to Vss, Red to Vdd, and White to I/O pin P15.



Board of Education Servo Port Jumper, Set to Vdd

Example Program: PingMeasureCmAndIn.bs2

This example BS2 program is an excerpt from Chapter 2 of the Stamps in Class text *Smart Sensors and Applications*. Additional PBASIC programs, one for the BS1 and another than runs on any model of BASIC Stamp 2 (BS2, BS2e, BS2sx, BS2p, BS2pe, BS2px) can be downloaded from the 28015 product page.

```
' Smart Sensors and Applications - PingMeasureCmAndIn.bs2
' Measure distance with Ping))) sensor and display in both in & cm

' {$STAMP BS2}
' {$PBASIC 2.5}

' Conversion constants for room temperature measurements.
CmConstant    CON   2260
InConstant    CON   890

cmDistance  VAR   Word
inDistance  VAR   Word
time        VAR   Word

DO

  PULSOUT 15, 5
  PULSIN 15, 1, time

  cmDistance = cmConstant ** time
  inDistance = inConstant ** time

  DEBUG HOME, DEC3 cmDistance, " cm"
  DEBUG CR, DEC3 inDistance, " in"

  PAUSE 100

LOOP
```

Propeller Microcontroller

```
{{
**************************************
*       Ping))) Object V1.1          *
*      (C) 2006 Parallax, Inc.       *
* Author:  Chris Savage & Jeff Martin *
* Started: 05-08-2006                *
**************************************

Interface to Ping))) sensor and measure its ultrasonic travel time.  Measurements can be in
units of time or distance.  Each method requires one parameter, Pin, that is the I/O pin that
is connected to the Ping)))'s signal line.

  ┌───────────────────┐
  │┌───┐         ┌───┐│    Connection To Propeller
  ││ ▮ │ PING))) │ ▮ ││    Remember PING))) Requires
  │└───┘         └───┘│    +5V Power Supply
  │    GND +5V SIG    │
  └─────┬───┬───┬─────┘
        │  ┌┘   ≷ 1K
        ∇  └┘   └┴ Pin

-------------------------REVISION HISTORY--------------------------
 v1.1 - Updated 03/20/2007 to change SIG resistor from 10K to 1K
}}

CON
  TO_IN = 73_746                                              ' Inches
  TO_CM = 29_034                                              ' Centimeters

PUB Ticks(Pin) : Microseconds | cnt1, cnt2
''Return Ping)))'s one-way ultrasonic travel time in microseconds

  outa[Pin]~                                                  ' Clear I/O Pin
  dira[Pin]~~                                                 ' Make Pin Output
  outa[Pin]~~                                                 ' Set I/O Pin
  outa[Pin]~                                                  ' Clear I/O Pin (> 2 µs pulse)
  dira[Pin]~                                                  ' Make I/O Pin Input
  waitpne(0, |< Pin, 0)                                       ' Wait For Pin To Go HIGH
  cnt1 := cnt                                                 ' Store Current Counter Value
  waitpeq(0, |< Pin, 0)                                       ' Wait For Pin To Go LOW
  cnt2 := cnt                                                 ' Store New Counter Value
  Microseconds := (||(cnt1 - cnt2) / (clkfreq / 1_000_000)) >> 1    ' Return Time in µs

PUB Inches(Pin) : Distance
''Measure object distance in inches

  Distance := Ticks(Pin) * 1_000 / TO_IN                       ' Distance In Inches

PUB Centimeters(Pin) : Distance
''Measure object distance in centimeters

  Distance := Millimeters(Pin) / 10                            ' Distance In Centimeters

PUB Millimeters(Pin) : Distance
''Measure object distance in millimeters

  Distance := Ticks(Pin) * 10_000 / TO_CM                      ' Distance In Millimeters
```

The ping.spin object is used in an example project with the Parallax 4 x 20 Serial LCD (#27979) to display distance measurements. The complete Project Archive can be downloaded from the Propeller Object Exchange at http://obex.parallax.com.

```
_____________________________________
Parallax Propeller Chip Project Archive
_____________________________________

 Project :  "ping_demo"

Archived :  Tuesday, December 18, 2007 at 3:29:46 PM

    Tool :  Propeller Tool version 1.05.8


            ping_demo.spin
               |
               ├──Debug_Lcd.spin
               |     |
               |     ├──Serial_Lcd.spin
               |     |     |
               |     |     └──Simple_Serial.spin
               |     |
               |     └──Simple_Numbers.spin
               |
               └──ping.spin
```

Javelin Stamp Microcontroller

This class file implements several methods for using the PING))) sensor with the Javelin Stamp module.

```
package stamp.peripheral.sensor;

import stamp.core.*;


/**
 * This class provides an interface to the Parallax PING))) ultrasonic
 * range finder module.
 * <p>
 * <i>Usage:</i><br>
 * <code>
 *   Ping range = new Ping(CPU.pin0);       // trigger and echo on P0
 * </code>
 * <p>
 * Detailed documentation for the PING))) Sensor can be found at: <br>
 * http://www.parallax.com/detail.asp?product_id=28015
 * <p>
 *
 * @version 1.0 03 FEB 2005
 */
public final class Ping {

  private int ioPin;

  /**
   * Creates PING))) range finder object
   *
   * @param ioPin PING))) trigger and echo return pin
   */
  public Ping (int ioPin) {
    this.ioPin = ioPin;
  }


  /**
   * Returns raw distance value from the PING))) sensor.
   *
   * @return Raw distance value from PING)))
   */
  public int getRaw() {

    int echoRaw = 0;

    CPU.writePin(ioPin, false);              // setup for high-going pulse
    CPU.pulseOut(1, ioPin);                  // send trigger pulse
    echoRaw = CPU.pulseIn(2171, ioPin, true); // measure echo return

    // return echo pulse if in range; zero if out-of-range
    return (echoRaw < 2131) ? echoRaw : 0;
  }


  /*
   * The PING))) returns a pulse width of 73.746 uS per inch.  Since the
   * Javelin pulseIn() round-trip echo time is in 8.68 uS units, this is the
   * same as a one-way trip in 4.34 uS units.  Dividing 73.746 by 4.34 we
   * get a time-per-inch conversion factor of 16.9922 (x 0.058851).
   *
```

```
   * Values to derive conversion factors are selected to prevent roll-over
   * past the 15-bit positive values of Javelin Stamp integers.
   */

  /**
   * @return PING))) distance value in inches
   */
  public int getIn() {
    return (getRaw() * 3 / 51);              // raw * 0.058824
  }


  /**
   * @return PING))) distance value in tenths of inches
   */
  public int getIn10() {
    return (getRaw() * 3 / 5);               // raw / 1.6667
  }


  /*
   * The PING))) returns a pulse width of 29.033 uS per centimeter.  As the
   * Javelin pulseIn() round-trip echo time is in 8.68 uS units, this is the
   * same as a one-way trip in 4.34 uS units.  Dividing 29.033 by 4.34 we
   * get a time-per-centimeter conversion factor of 6.6896.
   *
   * Values to derive conversion factors are selected to prevent roll-over
   * past the 15-bit positive values of Javelin Stamp integers.
   */

  /**
   * @return PING))) distance value in centimeters
   */
  public int getCm() {
    return (getRaw() * 3 / 20);              // raw / 6.6667
  }


  /**
   * @return PING))) distance value in millimeters
   */
  public int getMm() {
    return (getRaw() * 3 / 2);               // raw / 0.6667
  }
}
```

This simple demo illustrates the use of the PING))) ultrasonic range finder class with the Javelin Stamp:

```
import stamp.core.*;
import stamp.peripheral.sensor.Ping;


public class testPing {

  public static final char HOME = 0x01;

  public static void main() {

    Ping range = new Ping(CPU.pin0);
    StringBuffer msg = new StringBuffer();

    int distance;
```

```
    while (true) {
      // measure distance to target in inches
      distance = range.getIn();

      // create and display measurement message
      msg.clear();
      msg.append(HOME);
      msg.append(distance);
      msg.append(" \"   \n");
      System.out.print(msg.toString());

      // wait 0.5 seconds between readings
      CPU.delay(5000);
    }
  }
}
```

Resources and Downloads

You can find additional resources for the PING))) sensor by searching the following product pages at www.parallax.com:

- Smart Sensors and Applications (a Stamps in Class text), #28029
- PING))) Mounting Bracket Kit – a servo-driven mount designed to attach to a Boe-Bot robot, #570-28015
- Extension cable with 3-in header, #805-00011 (10-in.) or #805-00012 (14-in.)

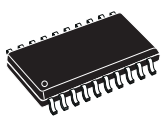
A video of a Boe-Bot robot using the PING))) sensor to scan its surroundings then drive to the closest object can be found under Resources > Video Library > Boe-Bot Robot Video Gallery.

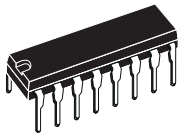
SGS-THOMSON MICROELECTRONICS

L293D
L293DD

PUSH-PULL FOUR CHANNEL DRIVER WITH DIODES

- 600mA OUTPUT CURRENT CAPABILITY PER CHANNEL
- 1.2A PEAK OUTPUT CURRENT (non repetitive) PER CHANNEL
- ENABLE FACILITY
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)
- INTERNAL CLAMP DIODES





SO(12+4+4) **Powerdip (12+2+2)**

ORDERING NUMBERS:

L293DD L293D

DESCRIPTION

The Device is a monolithic integrated high voltage, high current four channel driver designed to accept standard DTL or TTL logic levels and drive inductive loads (such as relays solenoides, DC and stepping motors) and switching power transistors.

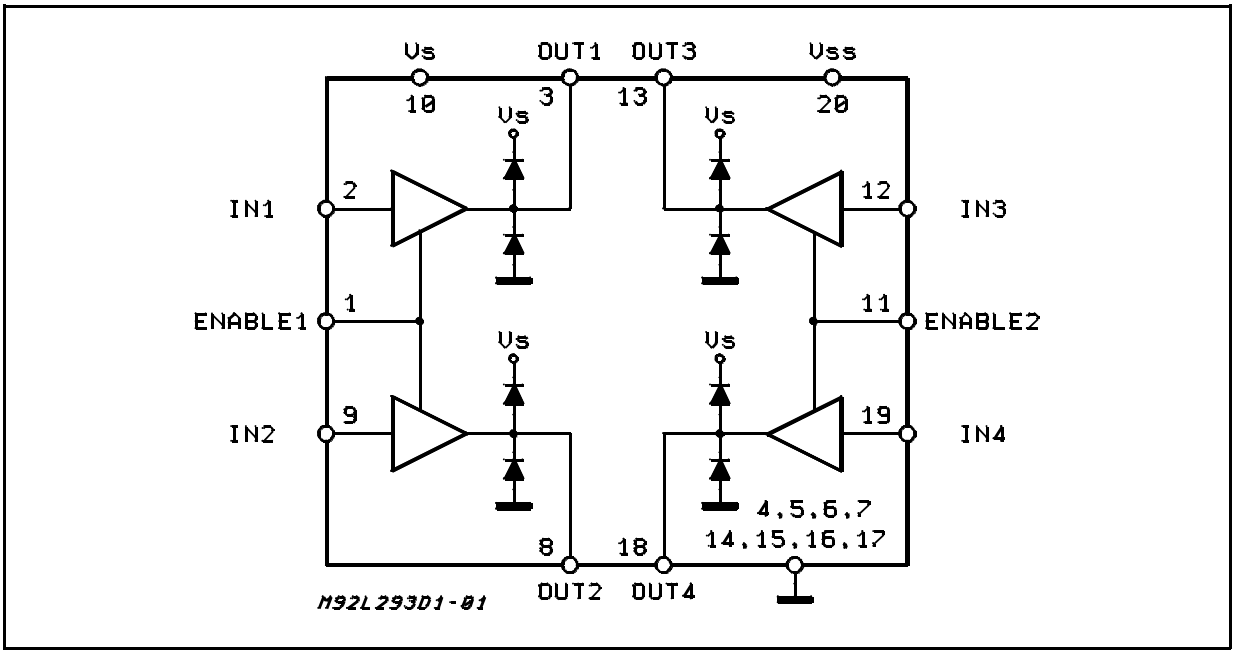
To simplify use as two bridges each pair of channels is equipped with an enable input. A separate supply input is provided for the logic, allowing operation at a lower voltage and internal clamp diodes are included.

This device is suitable for use in switching applications at frequencies up to 5 kHz.

The L293D is assembled in a 16 lead plastic packaage which has 4 center pins connected together and used for heatsinking

The L293DD is assembled in a 20 lead surface mount which has 8 center pins connected together and used for heatsinking.

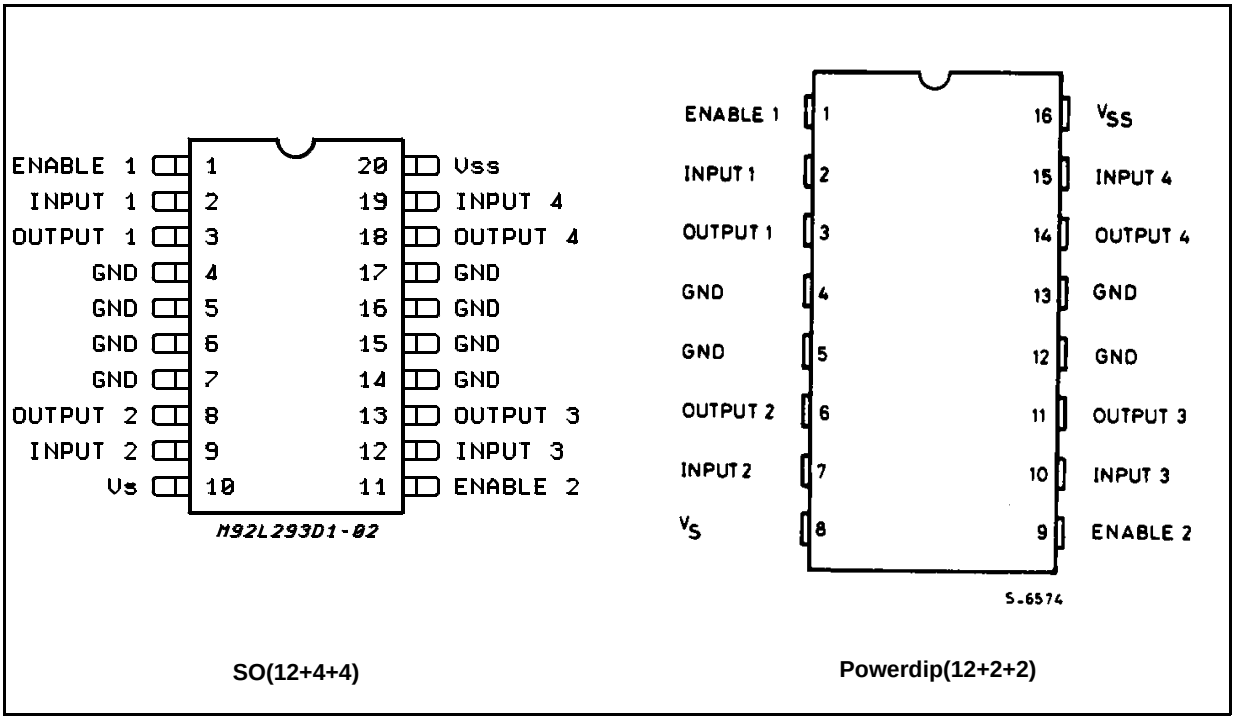
BLOCK DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Supply Voltage	36	V
V_{SS}	Logic Supply Voltage	36	V
V_i	Input Voltage	7	V
V_{en}	Enable Voltage	7	V
I_o	Peak Output Current (100 µs non repetitive)	1.2	A
P_{tot}	Total Power Dissipation at T_{pins} = 90 °C	4	W
T_{stg}, T_j	Storage and Junction Temperature	– 40 to 150	°C

PIN CONNECTIONS (Top view)



SO(12+4+4) **Powerdip(12+2+2)**

THERMAL DATA

Symbol	Decription		DIP	SO	Unit
$R_{th\ j\text{-}pins}$	Thermal Resistance Junction-pins	max.	–	14	°C/W
$R_{th\ j\text{-}amb}$	Thermal Resistance junction-ambient	max.	80	50 (*)	°C/W
$R_{th\ j\text{-}case}$	Thermal Resistance Junction-case	max.	14	–	

(*) With 6sq. cm on board heatsink.

ELECTRICAL CHARACTERISTICS (for each channel, V_S = 24 V, V_{SS} = 5 V, T_{amb} = 25 °C, unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_S	Supply Voltage (pin 10)		V_{SS}		36	V
V_{SS}	Logic Supply Voltage (pin 20)		4.5		36	V
I_S	Total Quiescent Supply Current (pin 10)	V_i = L ; I_O = 0 ; V_{en} = H		2	6	mA
		V_i = H ; I_O = 0 ; V_{en} = H		16	24	mA
		V_{en} = L			4	mA
I_{SS}	Total Quiescent Logic Supply Current (pin 20)	V_i = L ; I_O = 0 ; V_{en} = H		44	60	mA
		V_i = H ; I_O = 0 ; V_{en} = H		16	22	mA
		V_{en} = L		16	24	mA
V_{IL}	Input Low Voltage (pin 2, 9, 12, 19)		– 0.3		1.5	V
V_{IH}	Input High Voltage (pin 2, 9, 12, 19)	$V_{SS} \leq 7$ V	2.3		V_{SS}	V
		$V_{SS} > 7$ V	2.3		7	V
I_{IL}	Low Voltage Input Current (pin 2, 9, 12, 19)	V_{IL} = 1.5 V			– 10	µA
I_{IH}	High Voltage Input Current (pin 2, 9, 12, 19)	2.3 V ≤ V_{IH} ≤ V_{SS} – 0.6 V		30	100	µA
$V_{en\ L}$	Enable Low Voltage (pin 1, 11)		– 0.3		1.5	V
$V_{en\ H}$	Enable High Voltage (pin 1, 11)	$V_{SS} \leq 7$ V	2.3		V_{SS}	V
		$V_{SS} > 7$ V	2.3		7	V
$I_{en\ L}$	Low Voltage Enable Current (pin 1, 11)	$V_{en\ L}$ = 1.5 V		– 30	– 100	µA
$I_{en\ H}$	High Voltage Enable Current (pin 1, 11)	2.3 V ≤ $V_{en\ H}$ ≤ V_{SS} – 0.6 V			± 10	µA
$V_{CE(sat)H}$	Source Output Saturation Voltage (pins 3, 8, 13, 18)	I_O = – 0.6 A		1.4	1.8	V
$V_{CE(sat)L}$	Sink Output Saturation Voltage (pins 3, 8, 13, 18)	I_O = + 0.6 A		1.2	1.8	V
V_F	Clamp Diode Forward Voltage	I_O = 600nA		1.3		V
t_r	Rise Time (*)	0.1 to 0.9 V_O		250		ns
t_f	Fall Time (*)	0.9 to 0.1 V_O		250		ns
t_{on}	Turn-on Delay (*)	0.5 V_i to 0.5 V_O		750		ns
t_{off}	Turn-off Delay (*)	0.5 V_i to 0.5 V_O		200		ns

(*) See fig. 1.

TRUTH TABLE (one channel)

Input	Enable (*)	Output
H	H	H
L	H	L
H	L	Z
L	L	Z

Z = High output impedance
(*) Relative to the considered channel

Figure 1: Switching Times

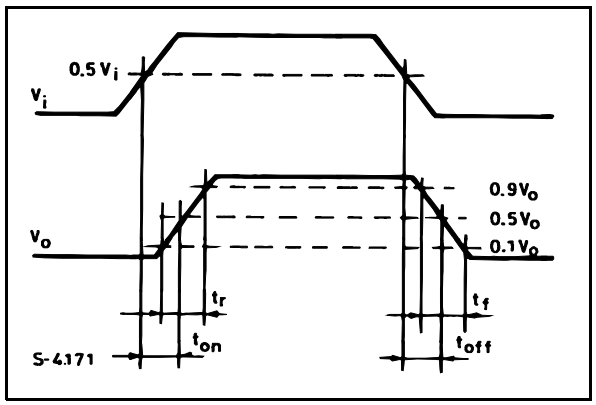
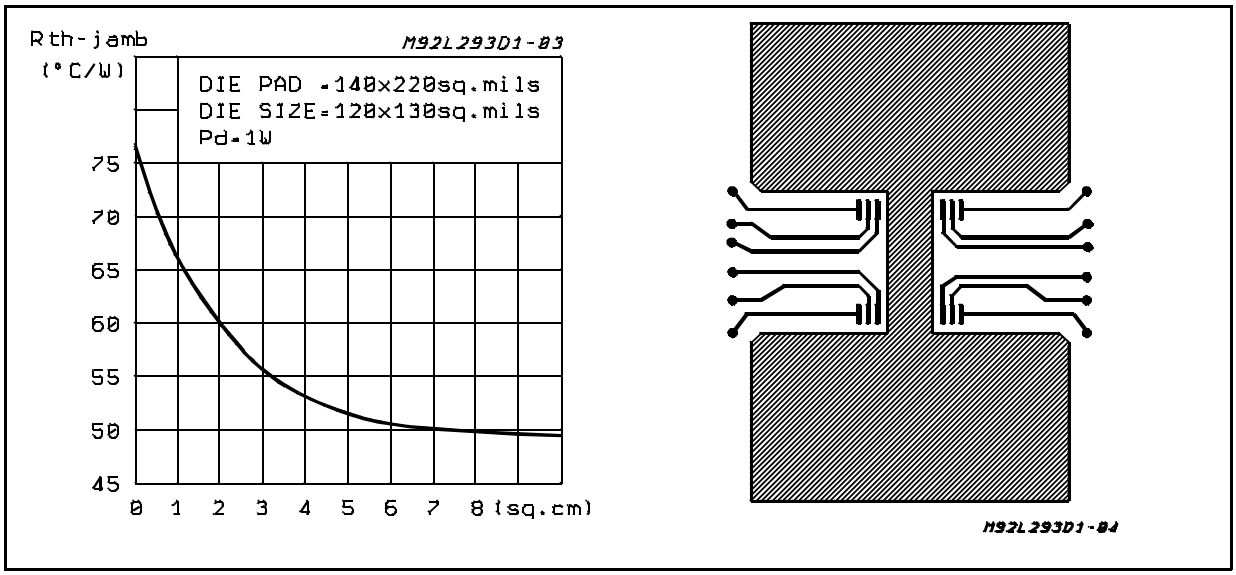
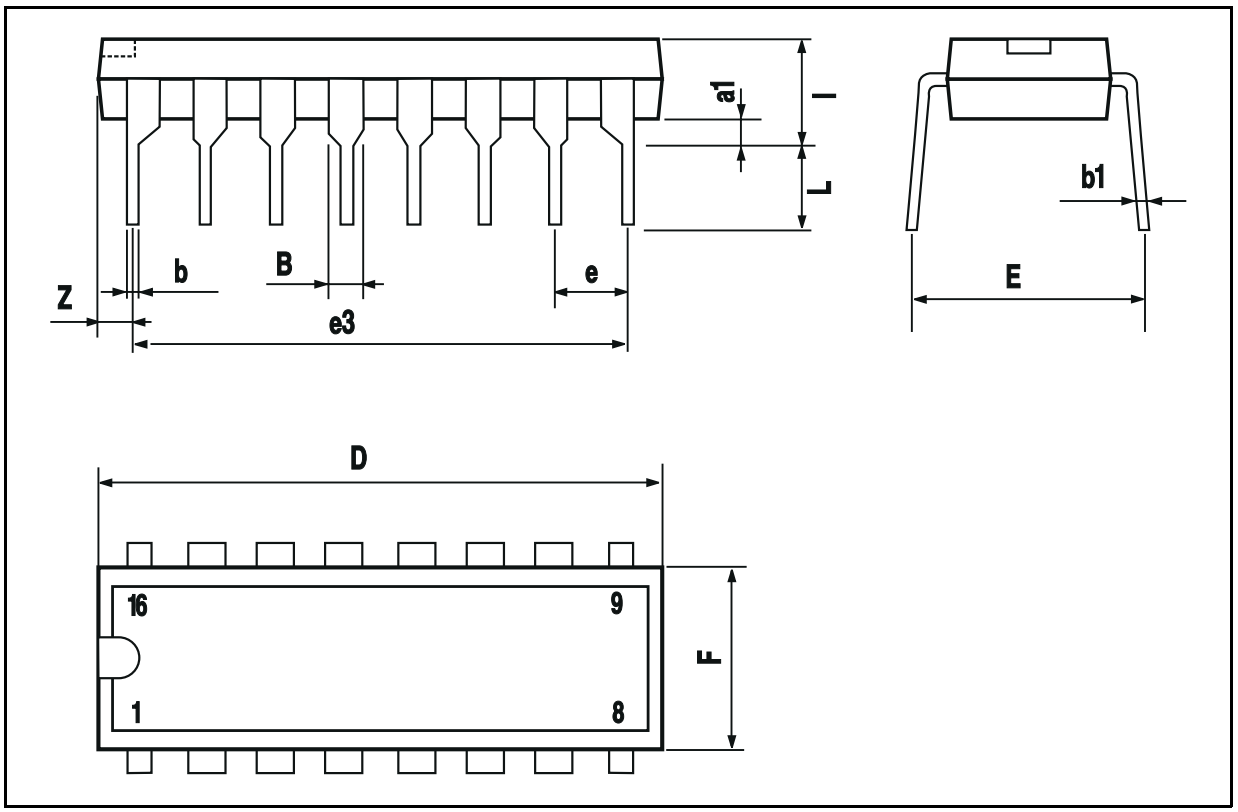


Figure 2: Junction to ambient thermal resistance vs. area on board heatsink (SO12+4+4 package)



POWERDIP16 PACKAGE MECHANICAL DATA

DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
a1	0.51			0.020		
B	0.85		1.40	0.033		0.055
b		0.50			0.020	
b1	0.38		0.50	0.015		0.020
D			20.0			0.787
E		8.80			0.346	
e		2.54			0.100	
e3		17.78			0.700	
F			7.10			0.280
I			5.10			0.201
L		3.30			0.130	
Z			1.27			0.050



SO20 PACKAGE MECHANICAL DATA

DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			2.65			0.104
a1	0.1		0.2	0.004		0.008
a2			2.45			0.096
b	0.35		0.49	0.014		0.019
b1	0.23		0.32	0.009		0.013
C		0.5			0.020	
c1		45			1.772	
D		1	12.6		0.039	0.496
E	10		10.65	0.394		0.419
e		1.27			0.050	
e3		11.43			0.450	
F		1	7.4		0.039	0.291
G	8.8		9.15	0.346		0.360
L	0.5		1.27	0.020		0.050
M			0.75			0.030
S	8° (max.)					

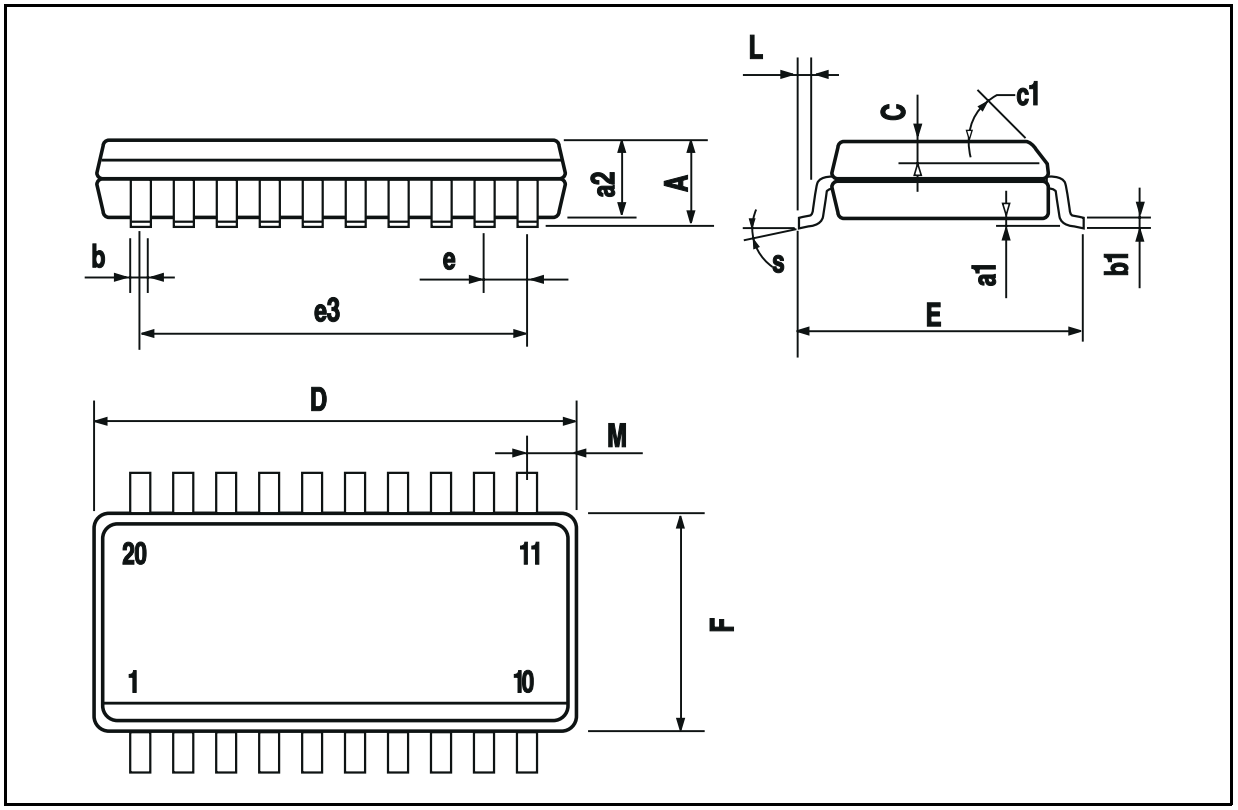


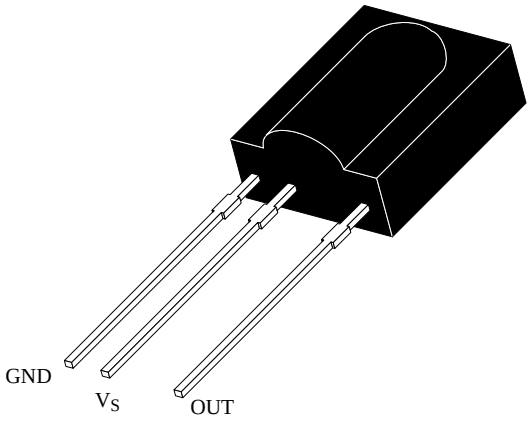
Photo Modules for PCM Remote Control Systems

Available types for different carrier frequencies

Type	fo	Type	fo
TSOP1730	30 kHz	TSOP1733	33 kHz
TSOP1736	36 kHz	TSOP1737	36.7 kHz
TSOP1738	38 kHz	TSOP1740	40 kHz
TSOP1756	56 kHz		

Description

The TSOP17.. – series are miniaturized receivers for infrared remote control systems. PIN diode and preamplifier are assembled on lead frame, the epoxy package is designed as IR filter.
The demodulated output signal can directly be decoded by a microprocessor. TSOP17.. is the standard IR remote control receiver series, supporting all major transmission codes.

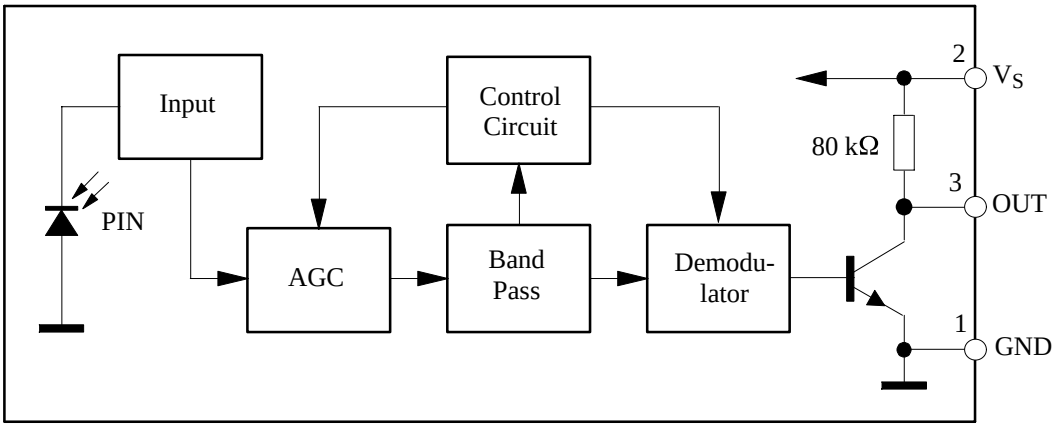


94 8691

Features

- Photo detector and preamplifier in one package
- Internal filter for PCM frequency
- Improved shielding against electrical field disturbance
- TTL and CMOS compatibility
- Output active low
- Low power consumption
- High immunity against ambient light
- Continuous data transmission possible (up to 2400 bps)
- Suitable burst length ≥10 cycles/burst

Block Diagram



94 8136

Absolute Maximum Ratings

T_{amb} = 25°C

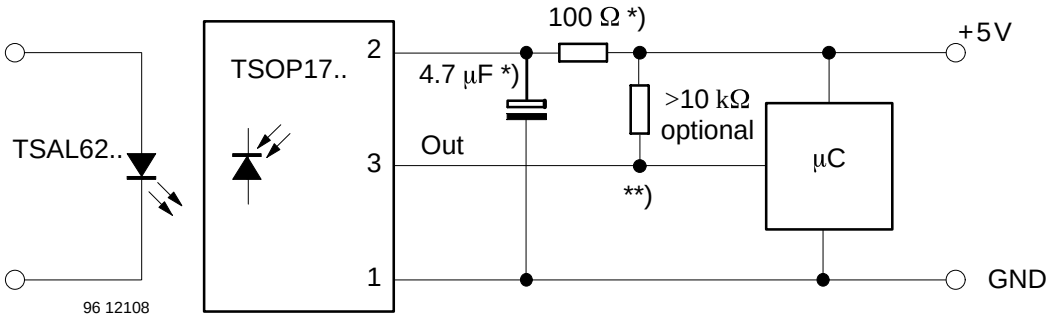
Parameter	Test Conditions	Symbol	Value	Unit
Supply Voltage	(Pin 2)	V_S	–0.3...6.0	V
Supply Current	(Pin 2)	I_S	5	mA
Output Voltage	(Pin 3)	V_O	–0.3...6.0	V
Output Current	(Pin 3)	I_O	5	mA
Junction Temperature		T_j	100	˚C
Storage Temperature Range		T_{stg}	–25...+85	˚C
Operating Temperature Range		T_{amb}	–25...+85	˚C
Power Consumption	(T_{amb} ≦ 85 ˚C)	P_{tot}	50	mW
Soldering Temperature	t ≦ 10 s, 1 mm from case	T_{sd}	260	˚C

Basic Characteristics

T_{amb} = 25°C

Parameter	Test Conditions	Symbol	Min	Typ	Max	Unit
Supply Current (Pin 2)	V_S = 5 V, E_v = 0	I_{SD}	0.4	0.6	1.5	mA
	V_S = 5 V, E_v = 40 klx, sunlight	I_{SH}		1.0		mA
Supply Voltage (Pin 2)		V_S	4.5		5.5	V
Transmission Distance	E_v = 0, test signal see fig.7, IR diode TSAL6200, I_F = 400 mA	d		35		m
Output Voltage Low (Pin 3)	I_{OSL} = 0.5 mA,E_e = 0.7 mW/m^2, f = f_o, t_p/T = 0.4	V_{OSL}			250	mV
Irradiance (30 – 40 kHz)	Pulse width tolerance: $t_{pi} - 5/f_o < t_{po} < t_{pi} + 6/f_o$, test signal (see fig.7)	$E_{e\ min}$		0.35	0.5	mW/m^2
Irradiance (56 kHz)	Pulse width tolerance: $t_{pi} - 5/f_o < t_{po} < t_{pi} + 6/f_o$, test signal (see fig.7)	$E_{e\ min}$		0.4	0.6	mW/m^2
Irradiance	$t_{pi} - 5/f_o < t_{po} < t_{pi} + 6/f_o$	$E_{e\ max}$	30			W/m^2
Directivity	Angle of half transmission distance	$\varphi_{1/2}$		±45		deg

Application Circuit



*) recommended to suppress power supply disturbances

**) The output voltage should not be hold continuously at a voltage below 3.3V by the external circuit.

Suitable Data Format

The circuit of the TSOP17.. is designed in that way that unexpected output pulses due to noise or disturbance signals are avoided. A bandpassfilter, an integrator stage and an automatic gain control are used to suppress such disturbances.
The distinguishing mark between data signal and disturbance signal are carrier frequency, burst length and duty cycle.

The data signal should fullfill the following condition:

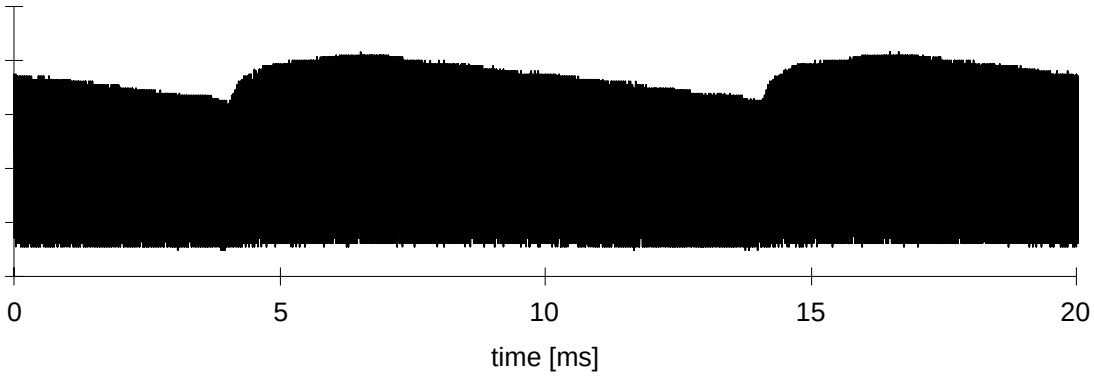
- Carrier frequency should be close to center frequency of the bandpass (e.g. 38kHz).
- Burst length should be 10 cycles/burst or longer.
- After each burst which is between 10 cycles and 70 cycles a gap time of at least 14 cycles is neccessary.
- For each burst which is longer than 1.8ms a corresponding gap time is necessary at some time in the data stream. This gap time should have at least same length as the burst.
- Up to 1400 short bursts per second can be received continuously.

Some examples for suitable data format are:
NEC Code, Toshiba Micom Format, Sharp Code, RC5 Code, RC6 Code, R–2000 Code, Sony Format (SIRCS).

When a disturbance signal is applied to the TSOP17.. it can still receive the data signal. However the sensitivity is reduced to that level that no unexpected pulses will occure.

Some examples for such disturbance signals which are suppressed by the TSOP17.. are:

- DC light (e.g. from tungsten bulb or sunlight)
- Continuous signal at 38kHz or at any other frequency
- Signals from fluorescent lamps with electronic ballast (an example of the signal modulation is in the figure below).



IR Signal from Fluorescent Lamp with low Modulation

Typical Characteristics (T_{amb} = 25°C unless otherwise specified)

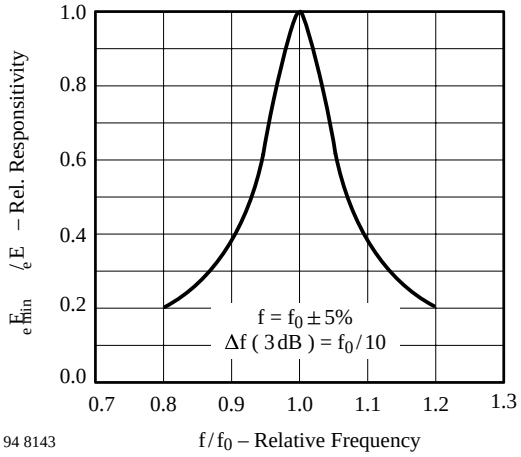


Figure 1. Frequency Dependence of Responsivity

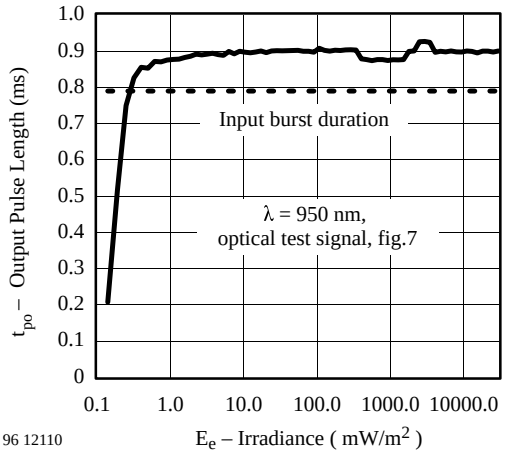


Figure 2. Sensitivity in Dark Ambient

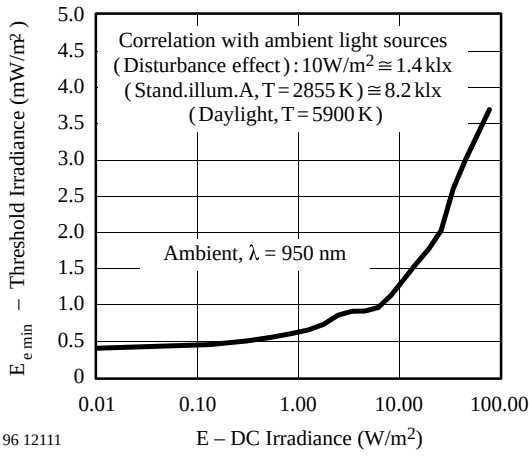


Figure 3. Sensitivity in Bright Ambient

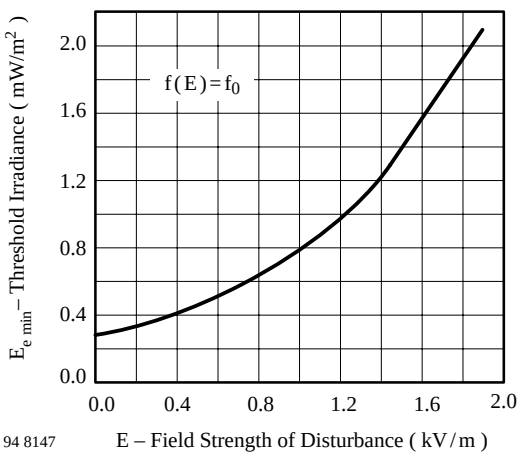


Figure 4. Sensitivity vs. Electric Field Disturbances

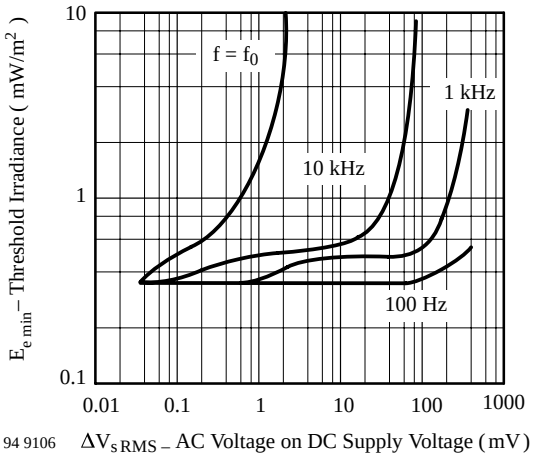


Figure 5. Sensitivity vs. Supply Voltage Disturbances

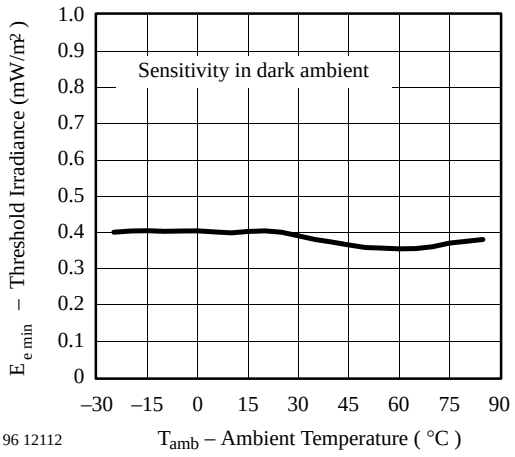


Figure 6. Sensitivity vs. Ambient Temperature

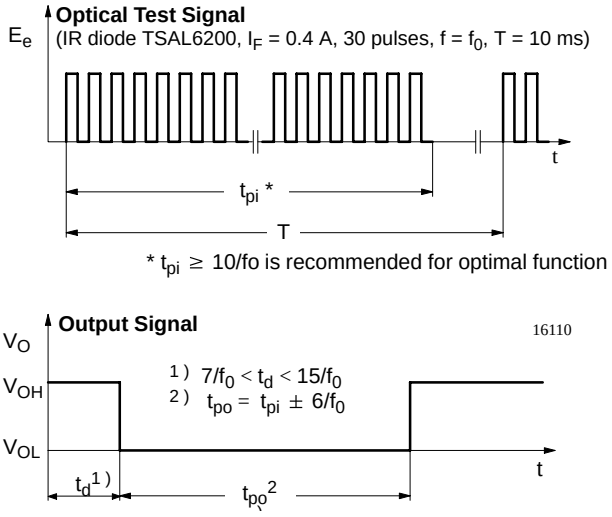


Figure 7. Output Function

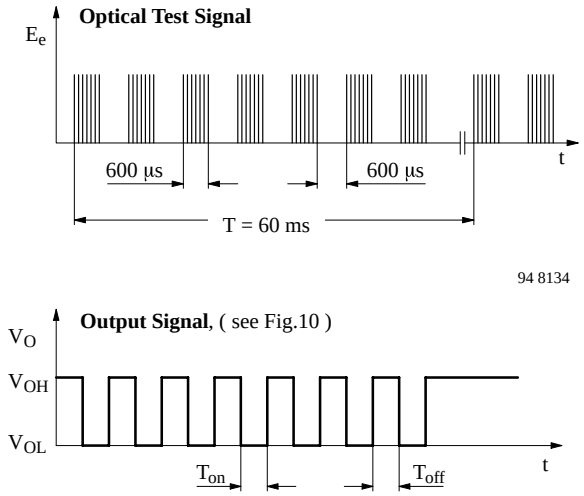


Figure 8. Output Function

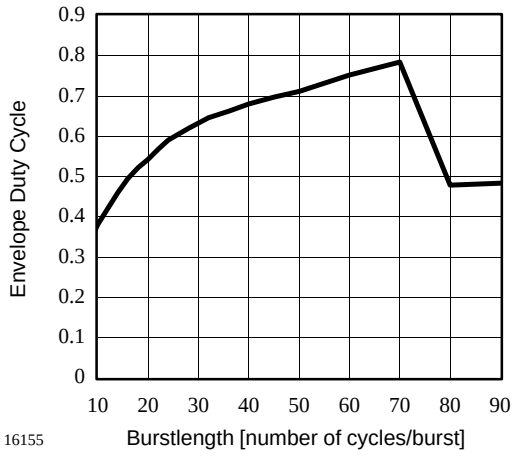


Figure 9. Max. Envelope Duty Cycle vs. Burstlength

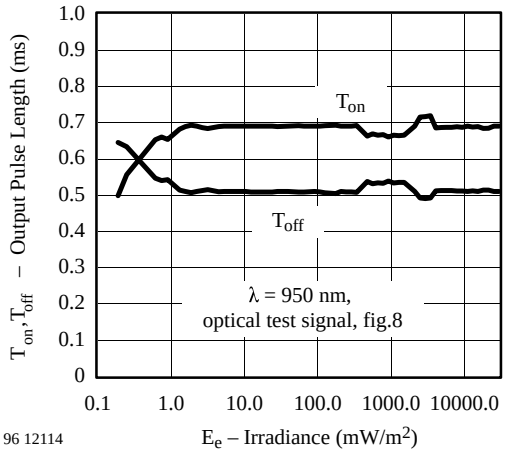


Figure 10. Output Pulse Diagram

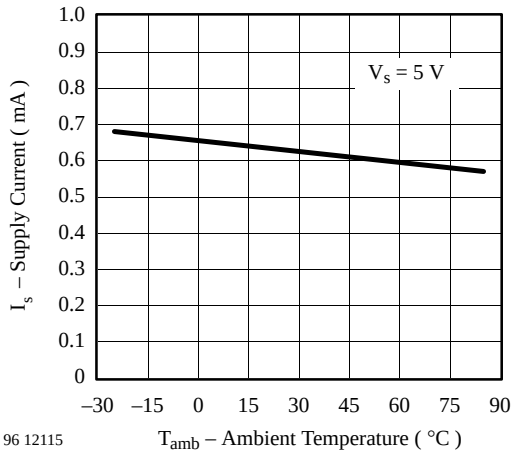


Figure 11. Supply Current vs. Ambient Temperature

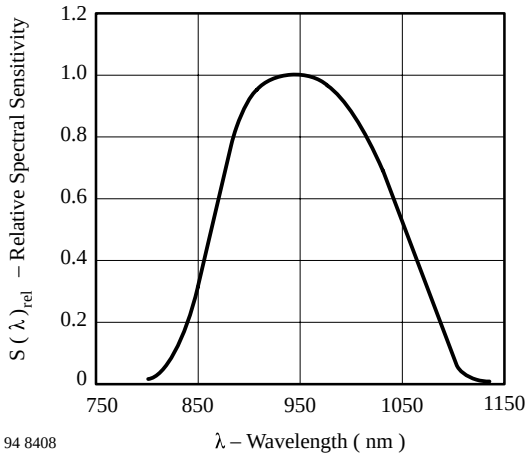


Figure 12. Relative Spectral Sensitivity vs. Wavelength

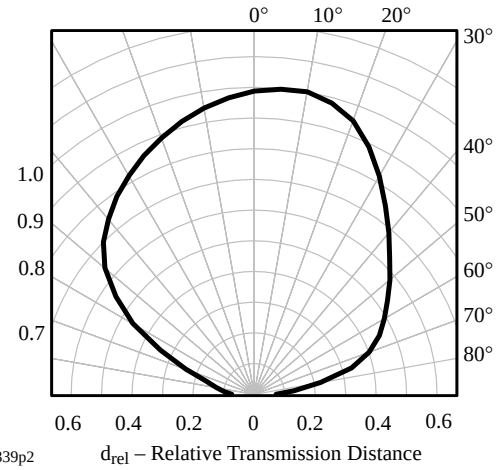


Figure 13. Vertical Directivity φ_y

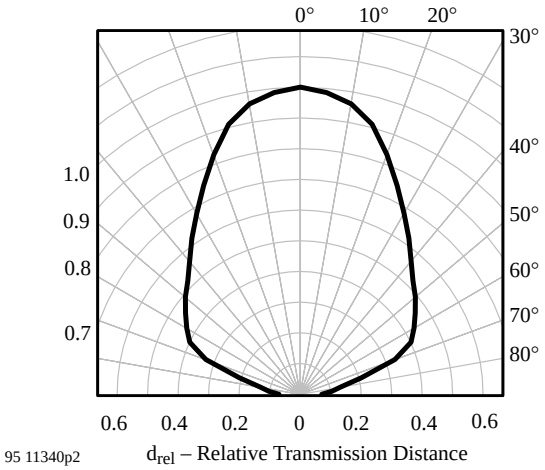
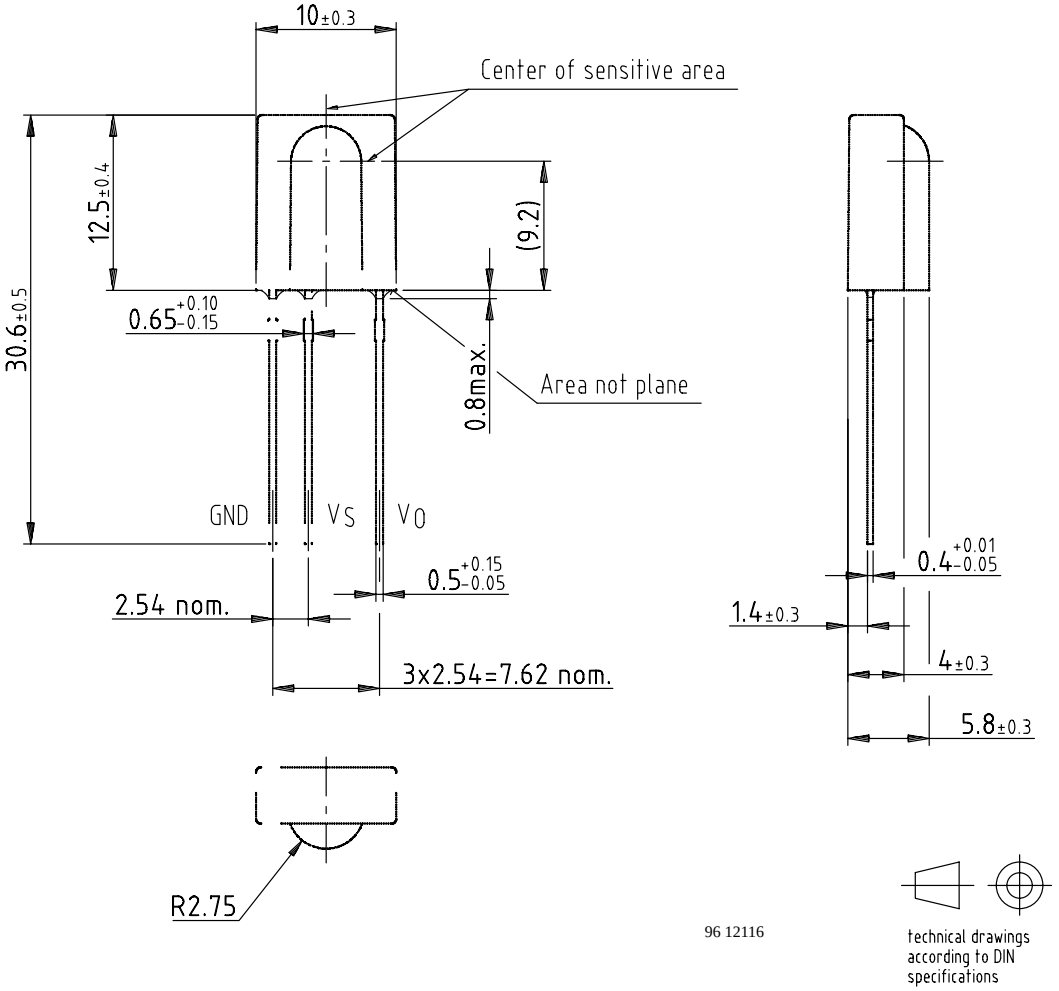


Figure 14. Horizontal Directivity φ_x

Dimensions in mm



Ozone Depleting Substances Policy Statement

It is the policy of **Vishay Semiconductor GmbH** to

1. Meet all present and future national and international statutory requirements.

2. Regularly and continuously improve the performance of our products, processes, distribution and operating systems with respect to their impact on the health and safety of our employees and the public, as well as their impact on the environment.

It is particular concern to control or eliminate releases of those substances into the atmosphere which are known as ozone depleting substances (ODSs).

The Montreal Protocol (1987) and its London Amendments (1990) intend to severely restrict the use of ODSs and forbid their use within the next ten years. Various national and international initiatives are pressing for an earlier ban on these substances.

Vishay Semiconductor GmbH has been able to use its policy of continuous improvements to eliminate the use of ODSs listed in the following documents.

1. Annex A, B and list of transitional substances of the Montreal Protocol and the London Amendments respectively

2. Class I and II ozone depleting substances in the Clean Air Act Amendments of 1990 by the Environmental Protection Agency (EPA) in the USA

3. Council Decision 88/540/EEC and 91/690/EEC Annex A, B and C (transitional substances) respectively.

Vishay Semiconductor GmbH can certify that our semiconductors are not manufactured with ozone depleting substances and do not contain such substances.

Vishay Semiconductor GmbH, P.O.B. 3535, D-74025 Heilbronn, Germany
Telephone: 49 (0)7131 67 2831, Fax number: 49 (0)7131 67 2423