République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mouloud Mammeri de Tizi-Ouzou
Faculté du Génie électrique et de l'Informatique
Département d'électronique

Mémoire de fin d'études

En vue de l'obtention du diplôme d'ingénieur d'état en électronique
Option : contrôle

Conception et réalisation d'une fraiseuse à commande numérique (CNC)

Réalisé par :
AKLI Mahdi.
GRIM Fayçal.
CHIKHI Ameziane.

Encadré par :
Mr. M.Laghrouche.

Promotion :

2011

Remerciements

****

Nos sincères remerciements vont à notre promoteur Mr. LAGHROUCHE ainsi que Amar, Hayat et à tous ceux qui nous ont assistés à la concrétisation de notre projet.

****

Dédicaces

Je dédie ce modeste travail à :

Mes très chers parents.

Mes frères wacyl et salim.

Ma sœur dida et ma belle sœur Lynda.

Mes oncles et toutes leurs familles.

Les mouh sa3, tout particulièrement : Mamine, Sofiane, Dahmane, Meriem.

Bernard et James qui m'ont soutenu tout au long de ce projet.

GRIM Fayçal

Je dédie ce modeste travail à :

Mes très chers parents, et à mes 2 petites sœurs hakima et fahima, à mes grands mères et à toute ma grande famille.

Tous mes amis.

A james et tony nafsouka.

CHIKHI Ameziane

Je dédie ce modeste travail à :

Mes très chers parents, mon frère Salim et toute ma famille.
Ma sœur Hayat et mon beau frère Amar sans lesquels ce projet n'aurai jamais vu le jour.
Notre petit ange adoré Sophia à qui je souhaite beaucoup de bonheur et de réussite dans la vie.
Toutes les personnes qui m'ont soutenu.
Mon meilleur ami Soso.
Tous mes amis (es).
Sans oublier lamine et meziane, qui ont pris l'initiative de se lancer dans l'aventure avec moi.

AKLI Mahdi

La théorie, C'est quand on sait tout et que rien ne fonctionne.
La pratique, c'est quand tout fonctionne et que personne ne sait pourquoi.

Albert EINSTEIN.

Chapitre III : L'INTERPRETEUR DE COMMANDES

Chapitre IV : REALISATION, TESTS ET APPLICATIONS PRATIQUES

Introduction Générale

Dans la société de consommation actuelle, la demande pour des produits innovants, de qualité, bons marchés ne cesse d'augmenter. Le contexte économique mondial est très concurrentiel et tous les secteurs d'activités des entreprises se doivent d'optimiser en permanence la qualité, les coûts et les délais. En effet, récemment de nombreuses délocalisations pour la production de produits à faible valeur ajoutée ont accompagné l'émergence des pays.

Par ailleurs, les récents progrès techniques de certains nouveaux pays industrialisés font que le segment des pièces à forte valeur ajoutée est désormais en concurrence. Le secteur de la production est au cœur de cette bataille économique et est donc très stratégique. Parmi les différents domaines de la production (conception, fabrication, contrôle, qualité, gestion des moyens et des ressources, maintenance, etc.), la fabrication par enlèvement de matière joue un rôle essentiel d'autant plus que ce procédé de fabrication est le plus répandu à ce jour. En effet, afin de répondre aux exigences des clients, la géométrie des pièces devient complexe et les spécifications dimensionnelles se resserrent. Malgré les progrès réalisés par les procédés primaires de mise en forme des matériaux (formage, fonderie, etc.), ils ne permettent que rarement l'obtention directe des surfaces fonctionnelles et l'usinage se révèle nécessaire à l'obtention des produits finaux.

Depuis plusieurs années, l'évolution des moyens électroniques et informatiques (automates programmables, ordinateurs embarqués et systèmes de contrôle, etc.) ont permis le développement des Machines-Outils à Commande Numérique (MOCN). Ces machines associées à la Conception et Fabrication Assistée par Ordinateur (CFAO) ont aidé à maintenir la compétitivité du procédé d'usinage, en augmentant la productivité et en améliorant la qualité. Des avancées technologiques doivent donc sans cesse être réalisées afin que les systèmes de production restent compétitifs.

Afin de présenter le travail réalisé permettant d'atteindre les objectifs fixés pour cette thèse, le mémoire s'articule de la façon suivante : Le chapitre I présentera la conception et la réalisation mécanique de notre machine, dans le deuxième nous allons présenter les différentes étapes menées pour la conception et la réalisation de la carte de commande et les protocoles et systèmes de communications. Dans le troisième chapitre nous allons présenter le logiciel d'interpréteur de commandes, et le dernier sera consacré à la réalisation, tests et applications pratiques.

I.1. Cahier des charges

Conception et réalisation d'une fraiseuse à commande numérique permettant le déplacement de la fraise sur trois axes (x, y, z).

I.2. Définition du fraisage

Le fraisage est un procédé qui réalise un état de surface par enlèvement progressif d'une certaine quantité de matière de la pièce usinée à un taux de mouvement ou d'avance relativement faible par une fraise tournant à une vitesse comparativement élevée.
La caractéristique principale du procédé de fraisage est l'enlèvement de matière sous forme de copeaux individuels par chaque dent.

I.3. Principe de fonctionnement de la machine

La machine a été conçue principalement en fer léger (châssis+support tige) assurant ainsi sa solidité et sa facilité d'assemblage et en bois dur (support fraiseuse + table de travail) pour sa légèreté. Elle comporte 3 axes (3 degrés de libertés x, y, z), suivant 3 tiges filetées ou chaque extrémité est reliée directement à l'axe d'un moteur pas à pas pour tourner avec ce dernier et ainsi engendrer le déplacement de la fraise.

Une fraiseuse à trois axes est constituée de trois chariots se déplaçant dans les trois directions désirées, c'est-à-dire la longueur, la largeur et la profondeur (X, Y, Z). Il est important que ces déplacements soient guidés de manière à ce que le chariot ne se déplace que sur une ligne. Pour arriver à ce résultat, nous devons utiliser des guides sur lesquels vont glisser les chariots. Ces guides vont empêcher tous les autres déplacements parasites qui viendraient amoindrir la précision de la machine.

Les guides utilisés sont montés sur des supports situés de chaque côté de l'axe. Il est important que ces guides aient un diamètre suffisant pour ne pas courber. Plus la machine est grande, plus le diamètre devra être important. Pour notre machine, des guides de 16mm de diamètre conviendront. Les guides doivent être parfaitement parallèles entre eux. Sinon le chariot se déplacera difficilement dessus et doivent aussi être dans une matière dure comme de l'acier trempé ou de l'inox trempé de manière à ne pas s'user par frottements.

Des capteurs de fin de course sont placés à chaque extrémité d'un axe, et celà pour éviter le blocage mécanique du chariot en cas d'une mauvaise programmation du fichier de commandes. Ces capteurs sont un moyen pratique pour permettre au chariot de se repérer et de se mettre à la position initiale.

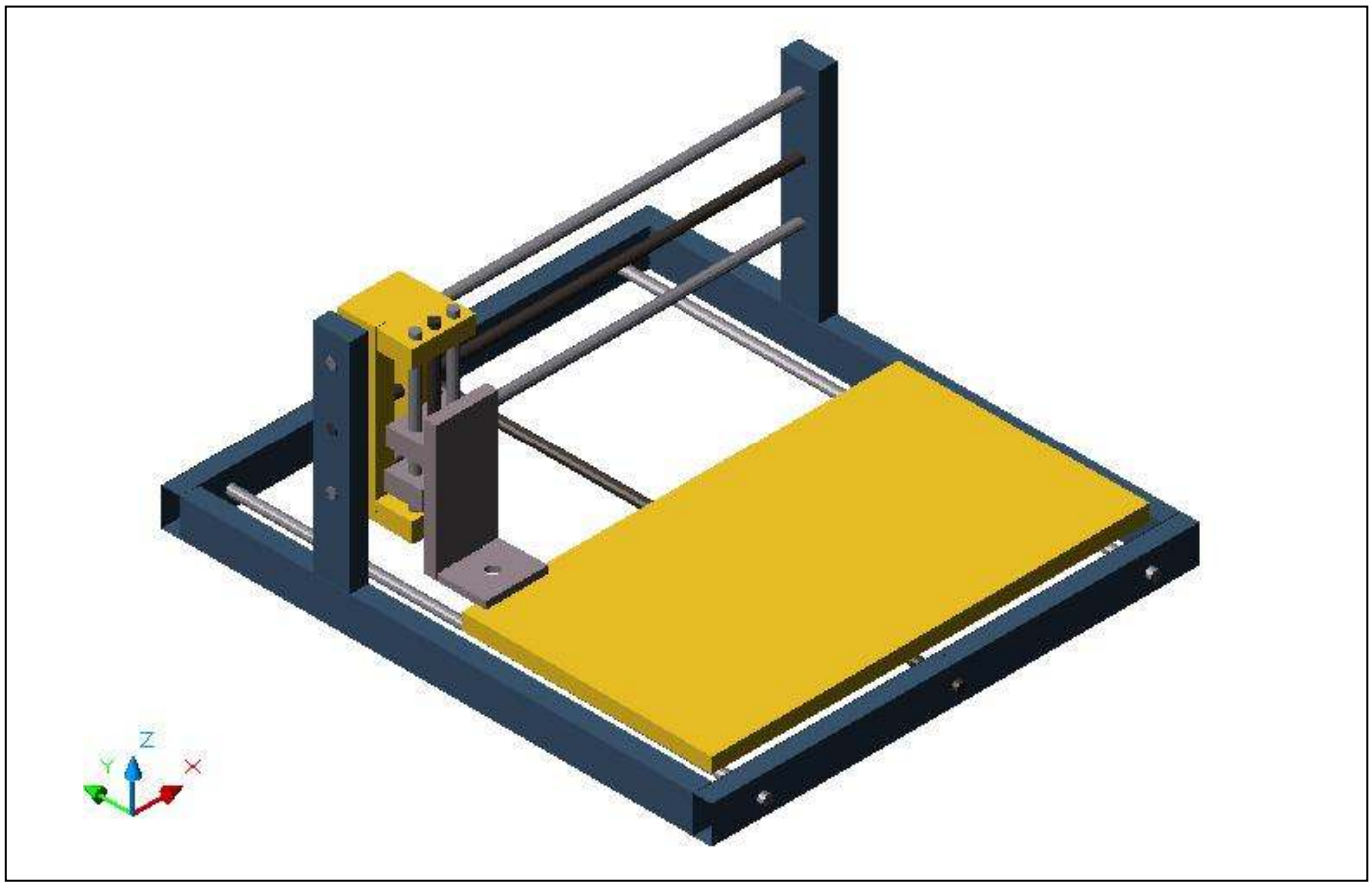


Fig. I.1 : Vue isométrique (S-O) de la fraiseuse.

Fig. I.1 : Vue isométrique (S-O) du chariot.

I.4. Dimensionnement de la machine

✓ **Les tiges filetées:**

Longueur (X) : 78 cm.
Largeur (Y) : 90 cm.
Hauteur (Z) : 29 cm.

✓ **La surface de travail :**

Longueur (X) 60 cm.
Largeur (Y) 40 cm.
Hauteur (Z) 8.5 cm.

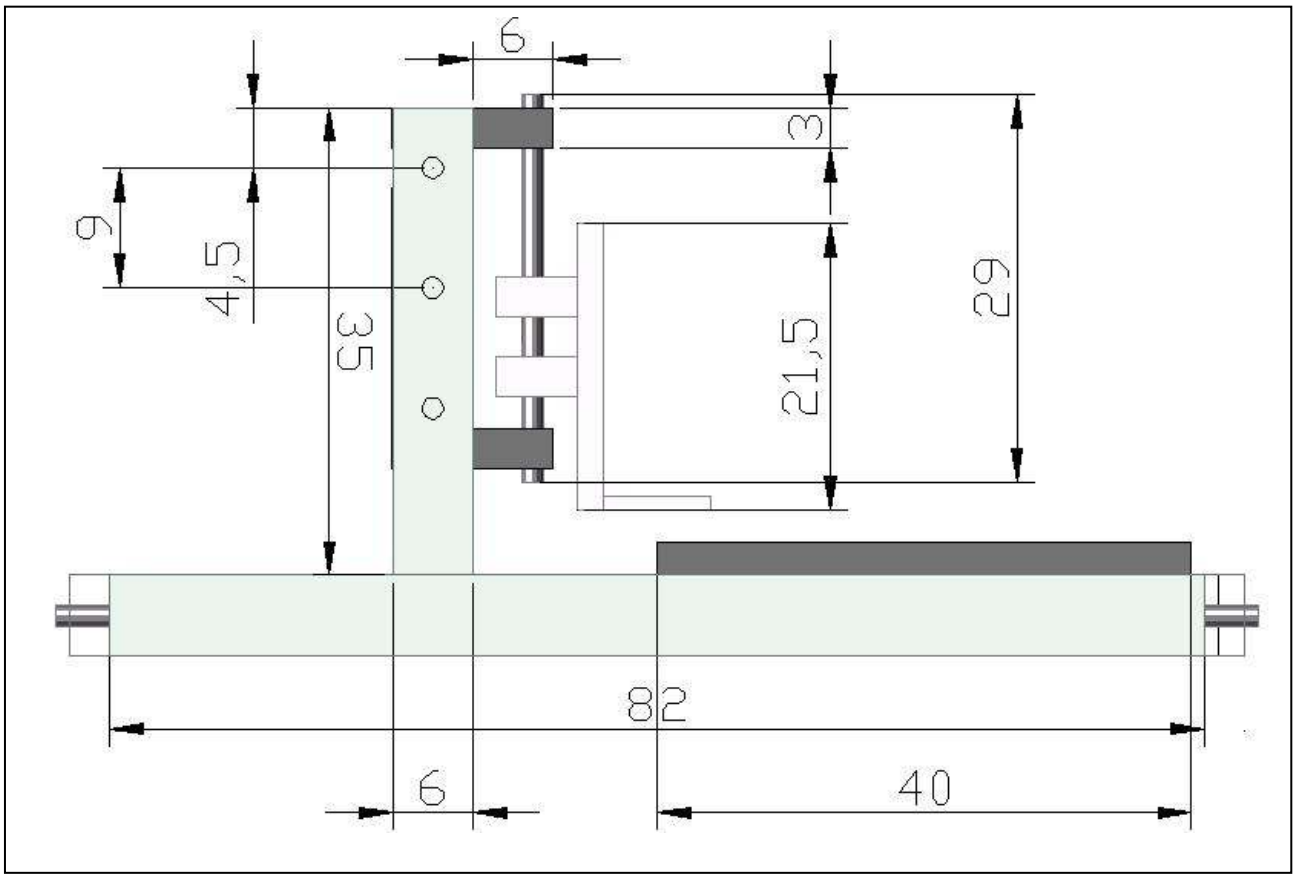


Fig. I.2 : vue latérale avec cotation (Unité : Cm).

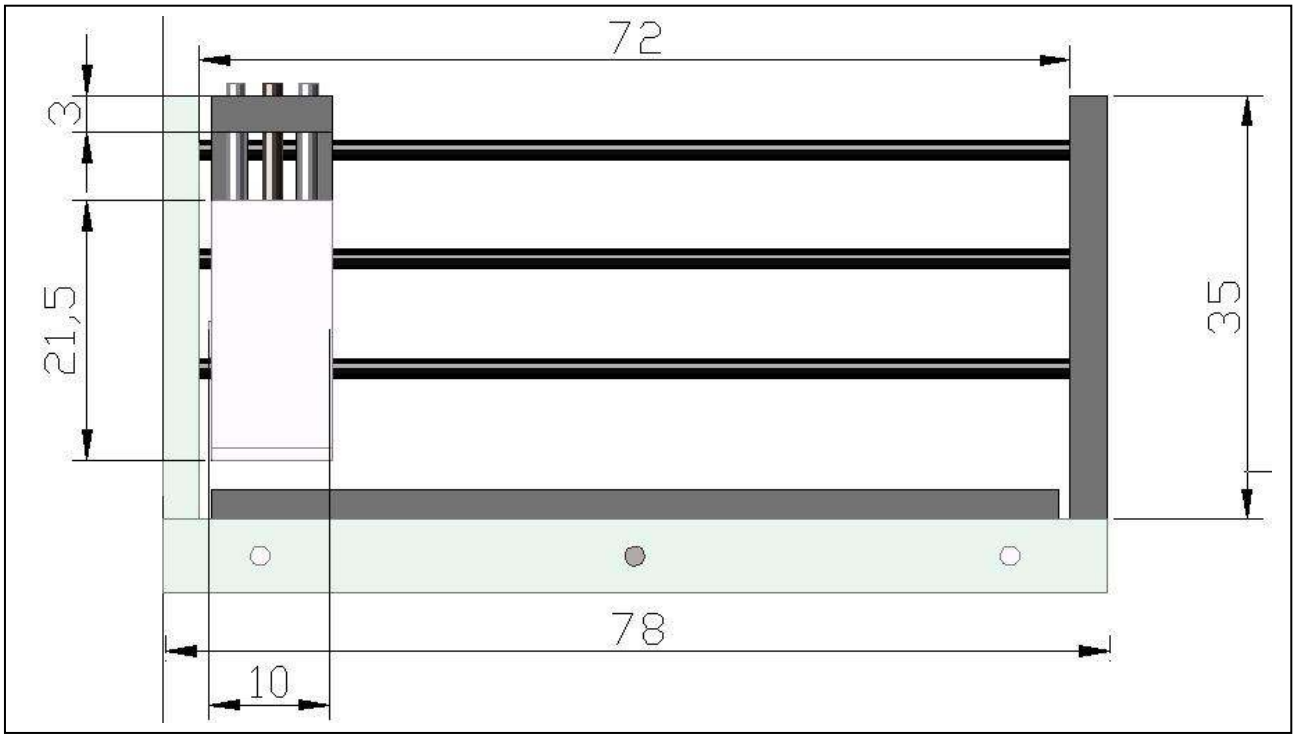


Fig. I.2 : vue avant avec cotation (Unité : Cm).

I.5. Choix des actionneurs

Les mouvements de la fraiseuse seront exécutés par des moteurs électriques pour des raisons évidentes de facilité de commande. Il s'imposait un choix entre les divers types de moteurs électriques existants (à courant continu, pas à pas, Servomoteurs, synchrone ou asynchrone).

S'il ne parait pas forcement évident de faire un choix parmi toutes les technologies, le prix et la facilité de mise en œuvre ont dicté notre choix. Les variateurs pour moteurs asynchrones et les commandes numériques capables de traiter un signal de position et de vitesse étant particulièrement hors de prix, le choix s'est porté de manière évidente vers le moteur pas à pas.

I.5.1. Définition d'un moteur pas à pas

Un moteur pas à pas est une machine tournante dont le rotor se déplace d'un angle élémentaire à chaque fois que son circuit de commande effectue une commutation de courant dans un ou plusieurs de ses enroulements, Il s'agit donc d'un actionneur de positionnement.
Les moteurs pas à pas sont différents des moteurs classiques. Au lieu de leur fournir une tension continue, on peut alimenter des bobines dans une séquence précise. Grâce à ce principe, l'angle de rotation peut être déterminé avec exactitude. De plus, en laissant une ou plusieurs bobines alimenter, on obtient un maintien (moteur figé).

Chaque impulsion envoyée par le circuit de commande au module de puissance se traduit par la rotation d'un pas du moteur. L'angle de rotation minimal entre deux modifications des impulsions électriques s'appelle un pas. On caractérise un moteur par le nombre de pas par tour (c'est à dire pour 360°). Les valeurs courantes sont 48, 100 ou 200 pas par tour.
Les impulsions électriques sont du type tout ou rien c'est à dire passage de courant ou pas.

Exemple:
Moteur à 400 pas=0.9°.
Moteur à 200 pas=1.8°.
Moteur à 100 pas=3.6°.
Moteur à 48 pas=7.5°.
Moteur à 24 pas=15°.

I.5.2. Les différents types de moteurs pas à pas

En peut classer les moteurs pas à pas en 3 catégories:

- ✓ Moteur à aimant permanent.
- ✓ Moteur à réluctance variable.
- ✓ Moteur hybride.

I.5.2.1. Moteur à aimant permanent

Il se compose de deux parties :

1. Le rotor qui est la partie mobile, comporte un nombre pair à aimant permanent magnétique dans le sens radial.
2. Le stator qui est la partie fixe, comporte d'encoches comportant 1,2, 3,4 enroulements Electriques.

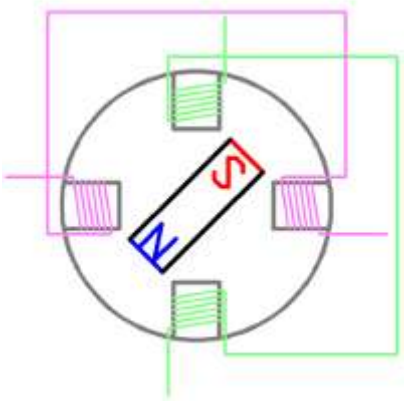


Fig. I.2: Moteur à aimant permanent.

✓ **Caractéristiques principales :**

- Faibles résolution : nombre de pas/tour peu important.
- Couple d'utilisation plus élevé par rapport au moteur à reluctance variable.
- Présentation d'un couple résiduel lorsque le moteur est hors tension.

I.5.2.2. Moteur à reluctance variable

✓ Le stator présente un certain nombre de dents ayant un bobinage.

✓ Le rotor (en matériau magnétique) possède un nombre différent de dents, mais sans bobinage.
✓ Le rotor se positionne pour que la réluctance du circuit magnétique soit minimum.
✓ 12 pas par tour ou 30° par tour.
✓ Des séquences pour un tour complet.

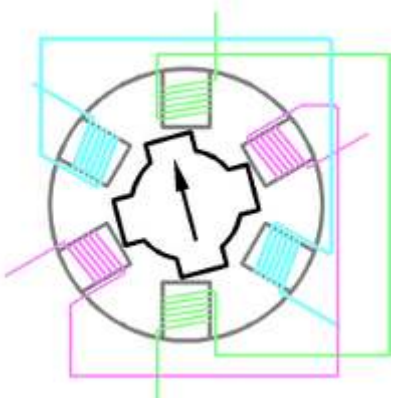


Fig. I.3: Moteur à reluctance variable.

✓ **Caractéristiques principales** :

- Les fréquences de fonctionnement peuvent être élevées.
- Bonne résolution.
- Construction simple mais délicate. Couple développé.
- Absence de couple résiduel avec le moteur hors tension.

I.5.2.3. Moteur hybride

C'est un moteur à reluctance polarisée.il superpose le principe de fonctionnement des moteurs à aimant permanent et à reluctance variable et combine leurs avantages.

Le rotor est constitué de deux disques dentés décalés mécaniquement entre ces deux disques et inséré un aimant permanent. Le stator et le rotor ont un nombre de dents paire de bobines, Le rotor place les dents nord et sud de telle façon que le flux traversant le rotor soit maximal.

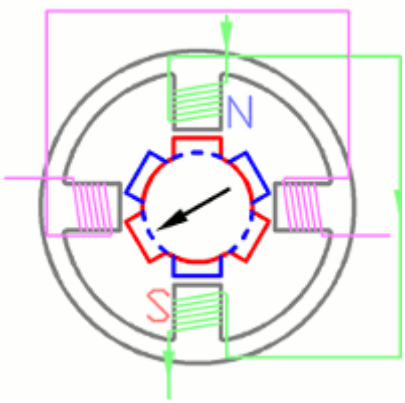


Fig. I.4: Moteurs hybride.

✓ **Caractéristiques principales** :

- Hors tension, le rotor est maintenu en position.
- Bonne précision de la position du rotor.
- Grande vitesse de rotation.

I. 6. Caractéristiques couple et vitesse

✓ Couple d'arrêt ou couple de maintien

C'est le couple maximum de rotation avec lequel on peut solliciter l'arbre d'un moteur pas à pas excité statiquement, sans qu'il ne se produise de modification de son angle de rotation.

✓ Plage de démarrage

C'est la plage dans laquelle un moteur pas à pas peut être actionné en synchronisation avec la Fréquence de travail sans rampe d'accélération ou de décélération.

✓ Fréquence limite de démarrage

C'est une fréquence maximale avec laquelle un moteur pas à pas ne peut démarrer à la charge indiquée.

✓ Plage d'accélération

C'est la plage de travail dans laquelle un moteur pas à pas peut être actionné en synchronisation avec la fréquence de travail, sans qu'il ne se produise d'erreur de pas. Il faut cependant qu'il soit actionné avec une rampe d'accélération et de décélération.

✓ Couple limite de travail ou d'entraînement

C'est un couple de rotation maximale avec lequel on peut solliciter un arbre de rotation avant qu'il ne sorte de la cadence.

✓ Fréquence maximale des pas

C'est une fréquence maximale admise avec laquelle un moteur pas à pas est actionné à vide sans perte de pas. Cependant, le moteur ne peut être démarré ou stoppé avec cette fréquence sans perte de pas.

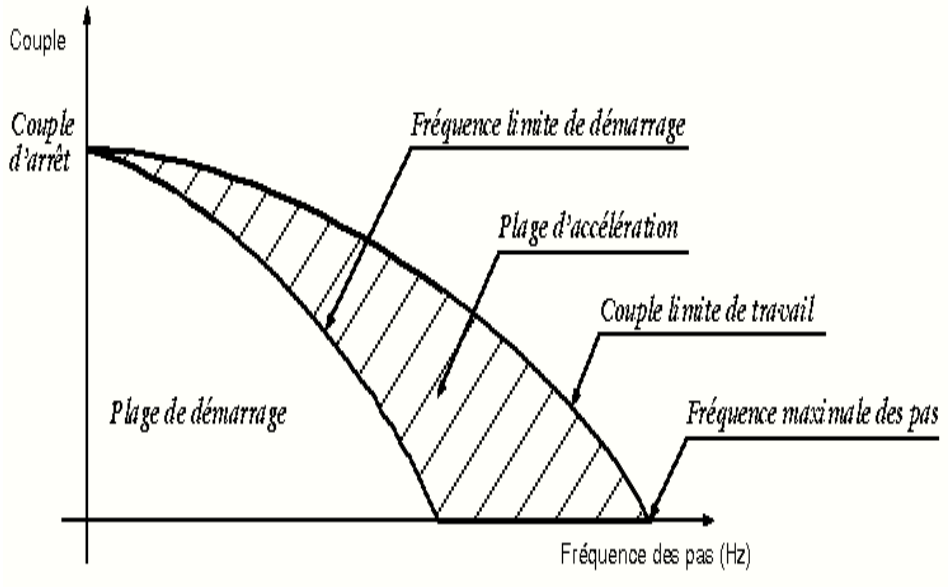


Fig. I.5 : Domaines de fonctionnement du moteur pas à pas.

I. 7. Configurations internes des bobines du moteur

I.7.1. Les moteurs à 4 fils (bipolaire)

Ce moteur agit comme s'il ne possédait que 2 bobines, il est obligé d'être alimenter soit une bobine à la fois, ou les deux en même temps. A tout moment, donc le moteur a la moitié ou la totalité de ses bobines alimentées, ce qui a comme avantage de lui donner plus de force.
Par contre, il est plus complexe de contrôler un moteur bipolaire, au niveau de l'interface de puissance.

I.7.2. Les moteurs à 5 fils (unipolaire)

Le moteur comporte deux bobines à point centrales, on relie le point central à l'alimentation et les autres bobines à l'interface de puissance.

I.7.3. Les moteurs à 6 fils

Avec le moteur à 6 fils, on a le choix d'une commande bipolaire, ou d'une commande unipolaire. Dans le premier cas, on ignore simplement les connexions centrales, et dans le second cas, on relie les deux points centraux au (+) de l'alimentation.

I.7.4. Les moteurs à 8 fils

Avec ce moteur, on a aussi le choix d'une commande bipolaire, ou unipolaire. Dans le premier cas, on ignore les 4 fils centraux, et dans le second cas, on reliant les quatre fils centraux ensemble.

I.8. Le moteur choisi

Les moteurs que nous avons choisis sont des moteurs bipolaires **2,6V**, **2×2A**, de taille **NEMA23 (57×57mm)** et de masse **500g**. Ils développent un couple nominal de **61Ncm**, ce qui satisfait parfaitement nos exigences. De plus, c'est des moteurs de **200 pas / tour** pouvant être commandés en mode demi pas, ce qui nous fait **400 pas / tour**.
Dès qu'il tourne, un moteur à besoin d'une tension supérieure à sa tension à l'arrêt si l'on veut maintenir son intensité nominale, garante du couple. Pour le bon fonctionnement en vitesse, c'est-à-dire pouvoir maintenir un certain couple à cette vitesse, on doit alimenter les moteurs avec une tension élevée égale au minimum à quatre fois la tension nominale du moteur.

L'utilisation d'une carte sans régulation nécessite l'ajout de résistances de puissance en série avec les bobines du moteur, ce qui implique une grande consommation électrique.
Ainsi, il est recommandé d'utiliser une carte avec régulation **PWM**, surtout quand la vitesse doit dépasser **2 tours / secondes**.

Une régulation **PWM** (Pulse Width Modulation), consiste à découper le courant envoyé dans un moteur en des tranches très fines, et de ne maintenir la tension que pendant une fraction du temps. Si l'on maintient durant la 'tranche', la tension pendant 33% du temps, le moteur se comportera comme s'il était alimenté avec une tension trois fois inférieure.

Une alimentation stabilisée n'est pas nécessaire pour les moteurs (avec utilisation de la PWM), du fait de la régulation continue du courant.

Les cartes de commande choisies pour nos moteurs sont à base du couple de circuit **(L297, L298)**, qui sont tout à fait appropriés pour ces moteurs.

Le circuit **L297** de **SGS-Thomson**, incorpore le système de séquençage, la régulation **PWM** et le contrôle de courant. Le courant ainsi haché, est envoyé aux moteurs à travers un étage de puissance (utilisation du **L298** dans notre cas).La carte de commande sera présentée dans le chapitre suivant.

II.1. Objectifs à atteindre

A ce stade, on dispose d'une structure mécanique qu'on doit commander par ordinateur, ce dernier envoie une information au PIC18F2550 (maitre) de la carte de commande principale via une communication USB qui sera transmise par la suite aux 3 pics moteurs 16F873A (esclave) via une communication I²C.

Cette information est sous forme d'octets et comporte le nombre de pas, la fréquence et le sens de rotation des moteurs.

Un traitement est exécuté au niveau de chaque pic moteur pour avoir un signal carré permettant la rotation des moteurs à travers la carte de commande moteur.

II.2. Structure électronique

La partie électronique de notre machine comporte deux types de cartes:

- ✓ Carte de commande principale.
- ✓ Carte de commande moteurs.

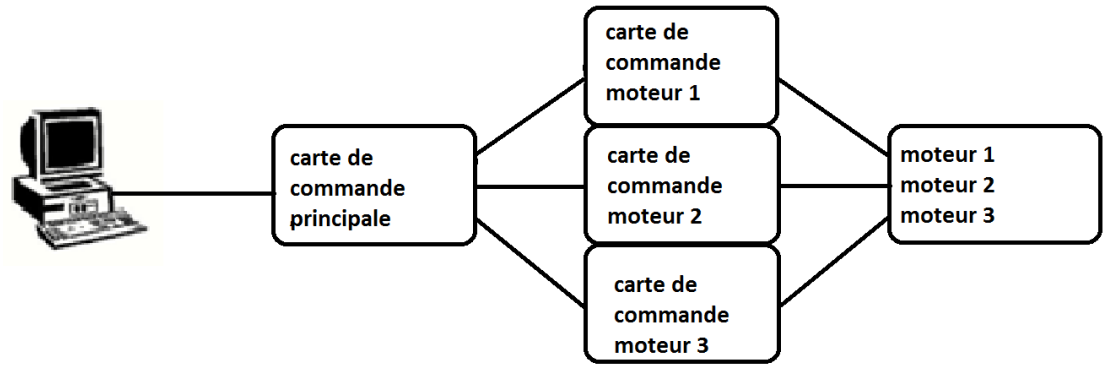


Fig. II.1: synoptique de notre réalisation

II.3 Carte de commande principale

Notre carte de commande est basée sur :

- ✓ Un PIC18F2550: qui assure la communication avec le PC via le bus USB et avec les Pics moteurs via le bus I²C.
- ✓ Trois Pics moteurs 16F873A : ils effectuent un calcul afin de piloter les cartes de commandes moteurs.

II.3.1 Présentation des PICS

✓ PIC18F2550

Ce modèle de PIC est un microcontrôleur de la famille High-end, fabriqué par la société Américaine Arizona MICROCHIP Technologie, cette famille a un jeu d'instructions plus complet puisqu'il comprend quelques 75 instructions. Cette palette d'instructions étendue lui permet de faire fonctionner du code C compilé de manière nettement plus efficace que les familles précédentes.

Le 18F2550 est un microcontrôleur pour des applications plus exigeantes ayant beaucoup de mémoire de programme (16k) et RAM (2k) et une interface USB complète - V2.0 (à basse vitesse 1,5 Mb/s et Full Speed 12 Mb/s).

✓ Brochage du PIC 18F2550

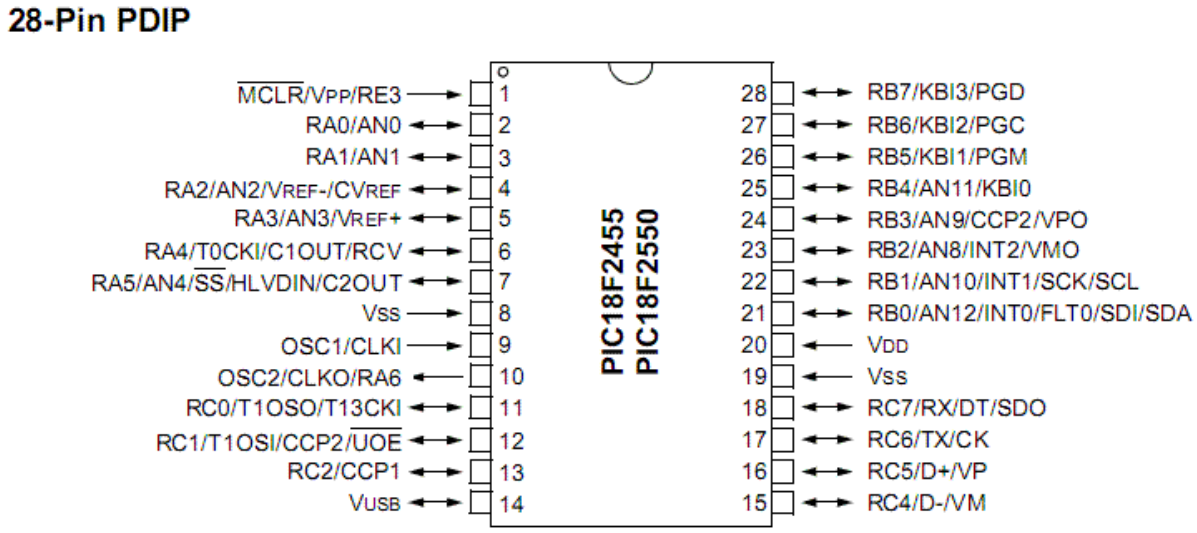


Fig. II.2: Brochage du PIC 18F2550

✓ **PIC 16F873A**

Le PIC16F873A est un microcontrôleur de type Mid-Range, il possède un jeu d'instructions réduit qui caractérise les circuits RISC (Reduced Instruction Set component).Les circuits RISC sont caractérisés par leurs rapidité d'exécution.

✓ **Brochage du PIC 16F873A**

Comme pour tout circuit intégré, chacune de ses broches a une ou plusieurs fonctions qui sont résumées par un sigle mnémotechnique.

La Fig. II.3 présente le brochage du PIC16F873A qui dispose de 28 broches montées sur un boîtier DIL.

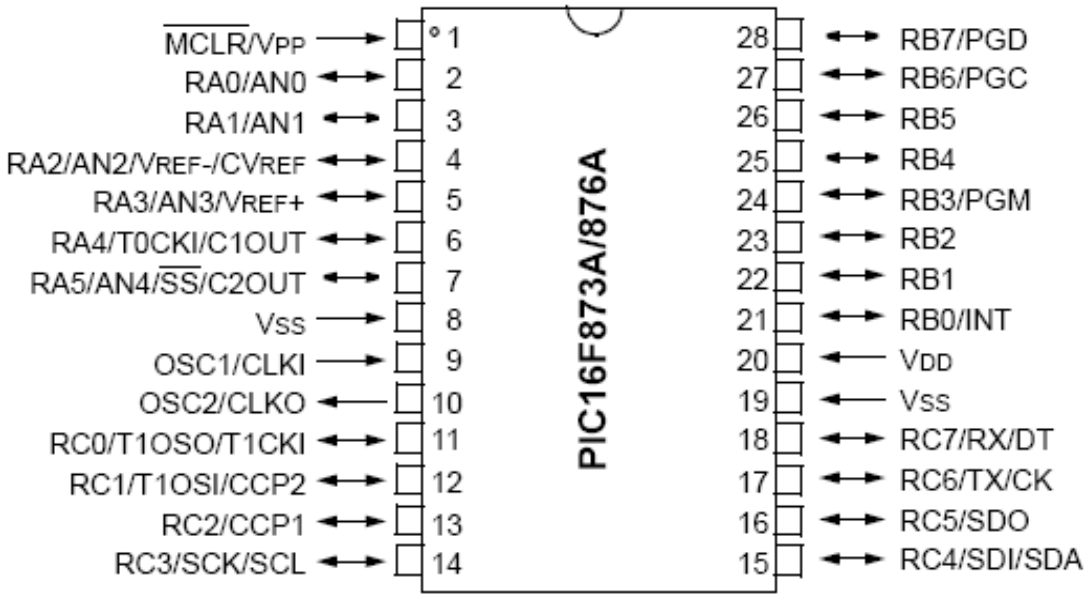


Fig. II.3: Brochage du PIC16F873A

II.3.2 Conception et réalisation de la carte de commande principale

II.3.2.1 Critères de choix des microcontrôleurs

Dans une opération de fraisage, la trajectoire de la fraise est d'une grande importance. D'ou l'utilisation d'un PIC pour chaque moteur s'avère indispensable dans le but de coordonner le mouvement des trois moteurs.

Le choix s'est porté sur le PIC16F873A puisqu'il est équipé du Bus I²c, une technologie de communication série sur deux fils intéressante dans notre cas vu la multitude de circuits intégrés.

Pour assurer la communication USB, notre choix s'est porté sur le PIC 18F2550 pour sa disponibilité.

II.3.2.2 Schéma électrique de la carte de commande principale

Puisque la carte de commande moteur dispose de Cinq entrées et en ajoutant les deux sorties des capteurs de fin de course, il faut prévoir l'utilisation de sept pins comme entrées/sorties (I/O) de chaque pic moteur.

La Fig. II.4 présente le schéma électrique de la carte de commande principale.

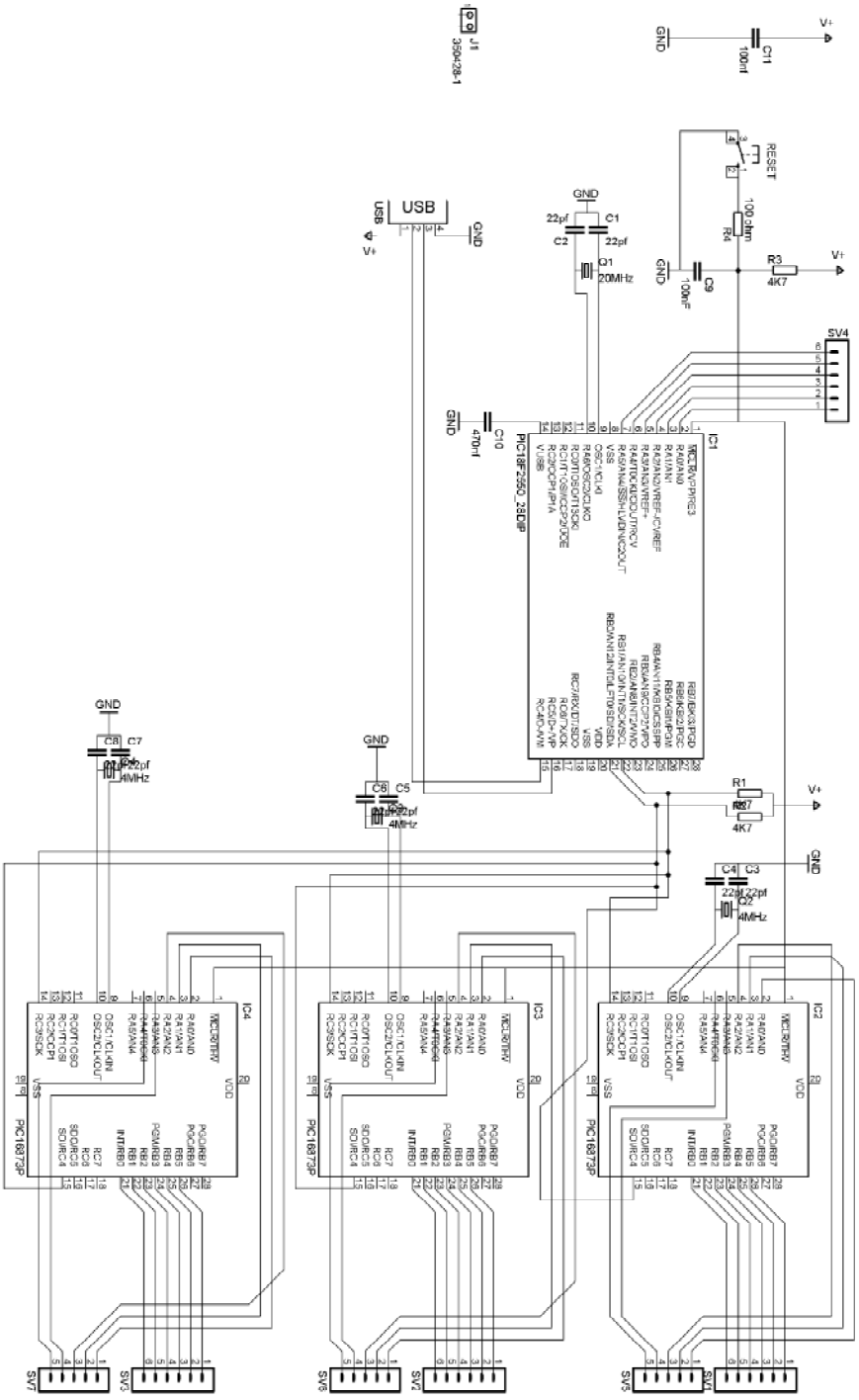


Fig. II.4 : schéma électrique de la carte de commande principale.

II.4. Protocoles et systèmes de communication

II.4.1. Principe de fonctionnement

A ce stade on dispose d'une électronique conçue pour pouvoir communiquer avec le software PC à travers le bus USB. Ce software doit être capable d'interpréter un fichier de commande écrit sous G-Code (norme RS274NGC) en données qui contiennent trois paramètres :

- ✓ La fréquence de rotation de chaque moteur.
- ✓ Le nombre de pas à effectuer par chaque moteur.
- ✓ Le sens de rotation de chaque moteur.

Cette communication est gérée par le microcontrôleur PIC18F2550 qui, une fois les données en provenance du pc reçues, procède à l'acheminement de chaque donnée correspondante vers le Pic moteur concerné a travers une communication I²c.

Le schéma de principe est donné en (Fig. II.5).

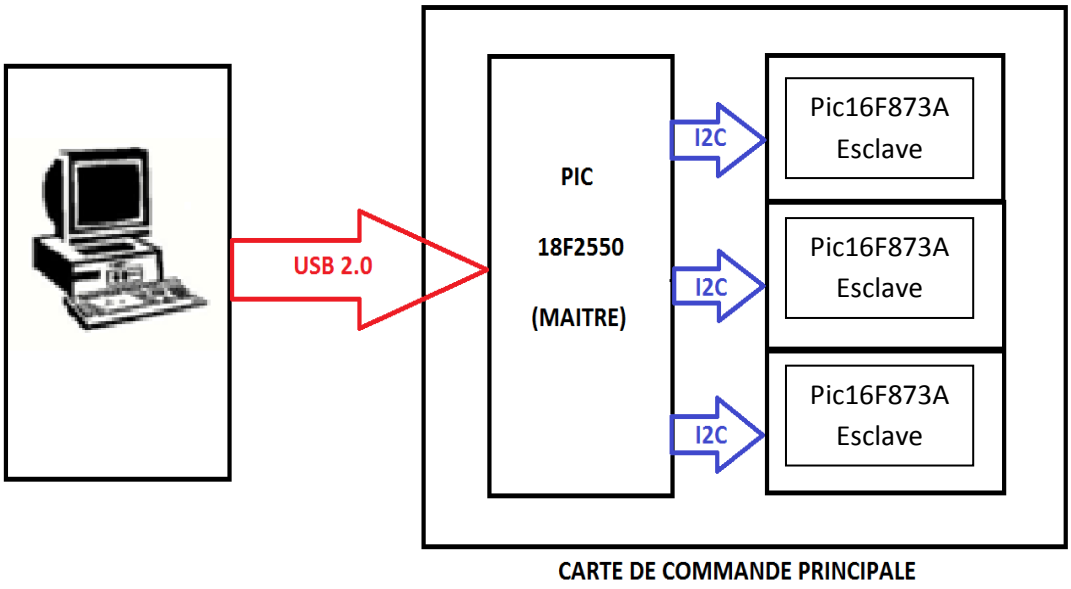


Fig. II.5 : schéma synoptique des systèmes de communication (USB, I2C)

II.4.2. La liaison USB

II.4.2.1. Le standard USB

L'USB (Universal Serial Bus) a été conçu pour remplacer les nombreux ports externes d'ordinateurs lents et incompatibles. C'est le successeur des antiques liaisons parallèles et RS-232, avec des performances nettement meilleures.

Ce bus a de grandes qualités et se décline en deux versions : une version transitoire l'USB 1.1 et depuis 2002 l'implantation stable qu'est la version USB 2.0.

Le bus USB est maintenant un standard sur tous les PC. Il est généralement utilisé pour brancher les imprimantes, scanners, modems et de nombreux appareils stockant des données (disque dur, clé USB…).

Nb : pour des raisons de marketing, il y a eu des changements de noms qui peuvent induire en erreur :

- ✓ L'ancienne norme USB 1.1 se nomme maintenant « USB 2.0 Full Speed », le débit atteint 12Mbits/s (1,5 Mo/s).
- ✓ La norme USB 2.0 devient « USB 2.0 High Speed », le débit atteint 480Mbits/s (60 Mo/s).

II.4.2.2. Description physique de l'USB

Le câble se compose de 4 fils et comporte un connecteur mâle de type A à une extrémité (connexion vers l'hôte) et un autre connecteur mâle de type A ou B à l'autre extrémité (connexion vers l'appareil). Un blindage est fortement recommandé pour une utilisation à 12 Mbits/s ou plus. La longueur maximale est de 5 mètres.

Au-delà d'une dizaine de mètres, la liaison devient défaillante à cause des retards "d'handshake", on perd alors des données lors de la transmission. (L'handshake est la séquence de signaux échangés entre deux appareils afin d'assurer la synchronisation de la transmission des données).

On retrouve ces deux connecteurs sur l'image ci dessous :

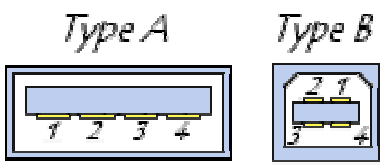


Fig. II.6 : Connecteurs USB.

Le câble utilisé est composé de quatre fils isolés :

- ✓ deux sont pour l'alimentation, un au potentiel +5V (VBUS) (qui permet d´alimenter éventuellement les périphériques USB) et l'autre à la masse GND.

- ✓ les deux fils restants DATA+ (D+) et DATA- (D-) forment une paire torsadée qui transfère les signaux de données différentiels. La transmission différentielle améliore l'immunité aux bruits parasites de l'environnement physique du périphérique ou de son câble.

N° de la broche	Couleur du fil	Fonction
n° 1	Rouge	Alimentation Vbus (5v)
n° 2	blanc	D-
n° 3	vert	D+
n° 4	noir	masse

Tab. II.1 : Brochage du connecteur USB.

La spécification USB impose une tension maximale de 5V sur les fils et un courant de 500 mA au plus. Il est évident que les fils d'alimentation sont toujours au même potentiel (tension de 5V si le périphérique est connecté, 0 sinon). Quant aux fils de données, la tension différentielle entre les deux peut valoir 0, 3.3 ou 5V.

II.4.2.3. Identification de la vitesse

UN appareil USB doit indiquer sa vitesse en mettant D+ ou D- à la tension 3.3V.

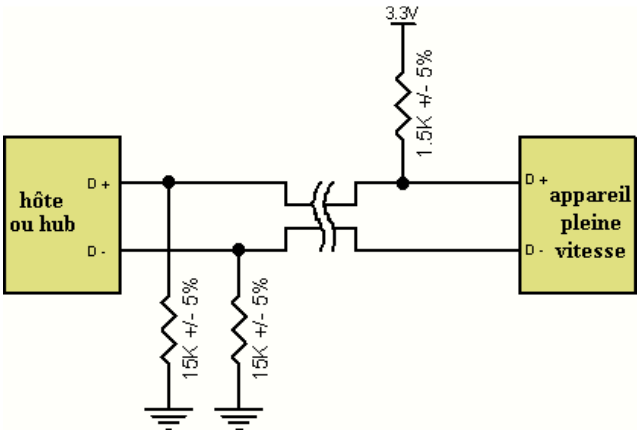


Fig. II.7. Appareil **plein vitesse** utilisant une résistance de rappel rattachée à D+.

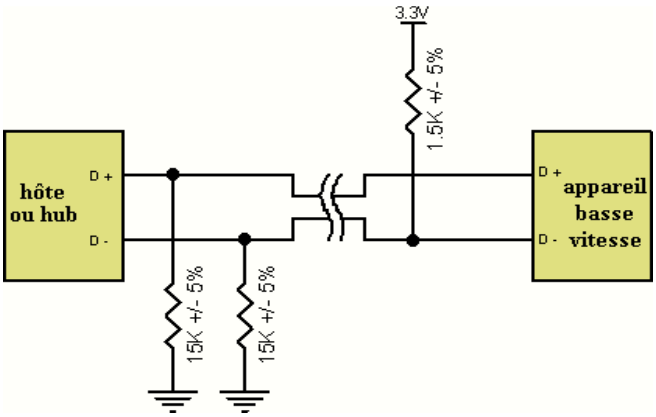


Fig. II.8. Appareil **bas vitesse** utilisant une résistance de rappel rattachée à D-.

II.4.3. Protocole de communication USB

La spécification USB impose un protocole de communication en plusieurs couches superposées, ce qui permet à l'utilisateur de ne manipuler que la ou les couches supérieures.

II.4.3.1. Schéma type des transactions

Les transactions USB se font par l'intermédiaire de l'émission de plusieurs paquets dont le format obéit à un standard. Chaque transaction consiste en la succession

- ✓ d'un paquet Jeton (Token).
- ✓ d'un paquet de données (DATA).
- ✓ d'un paquet d'état (HandShake).

Décrivons le schéma type d'une communication USB : Comme nous l'avons déjà vu, tout le bus est géré par l'hôte, ce qui signifie que c'est lui qui initie toutes les transactions en envoyant un paquet Jeton dans lequel figurent le type de transaction (lecture ou écriture), l'adresse du périphérique de destination, et la terminaison désignée. Suit le paquet DATA qui contient les informations réellement utiles dans la transaction, puis le paquet d'état qui indique si l'échange s'est correctement déroulé.

II.4.3.2. Les terminaisons

En fait, comme l'indique le schéma ci-dessous, chaque périphérique USB est décomposé en plusieurs sous blocs, possédant chacun un rôle différent dans la communication. Il s'agit ici de décrire de manière générale cette architecture, et non de développer complètement le fonctionnement précis de chaque étage.

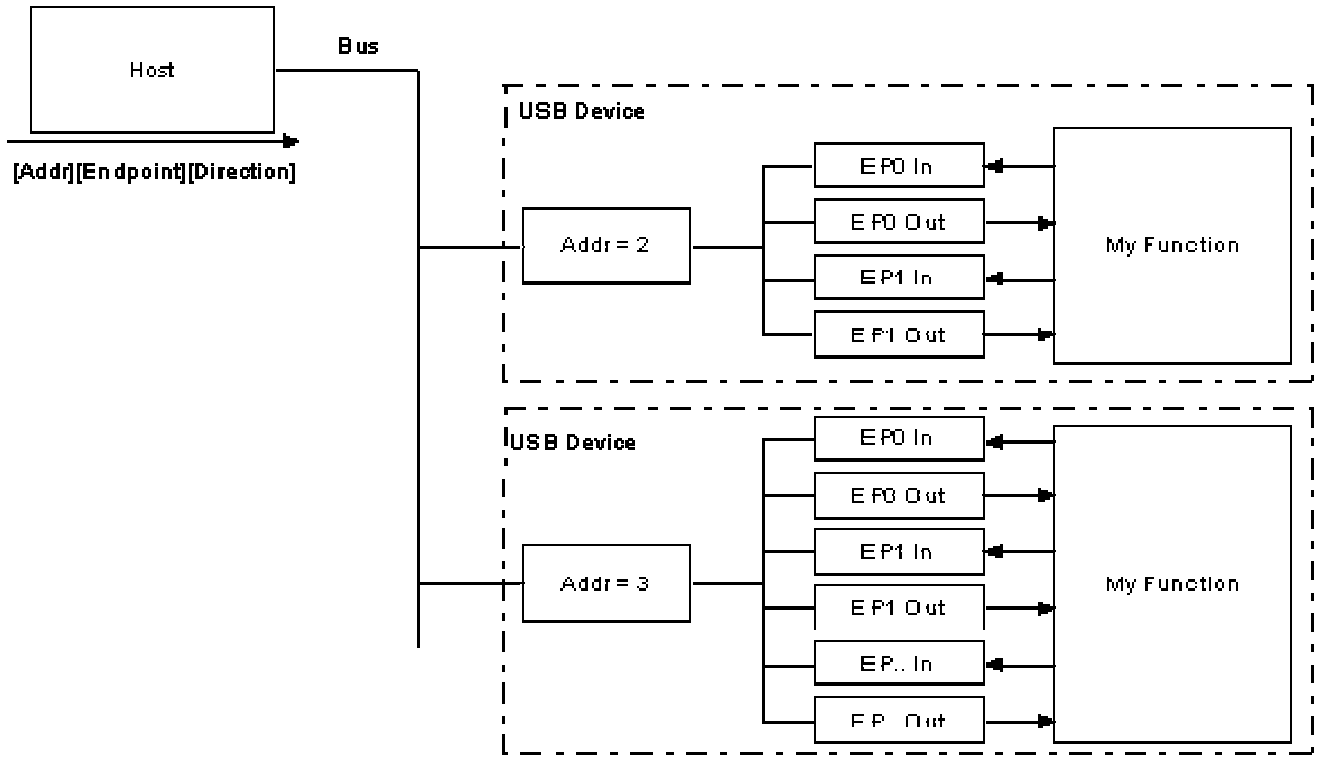


Fig. II.9 : Schéma du principe de fonctionnement des terminaisons USB.

Nous pouvons distinguer 3 sous blocs principaux :

- ✓ La partie qui décode l'adresse émise par l'hôte dans le paquet Jeton. Cette entité permet au périphérique de savoir que c'est bien à lui que l'hôte s'adresse.
- ✓ La partie terminaison.
- ✓ La partie réalisant la fonction USB proprement dit.

Les terminaisons peuvent être vues comme des intermédiaires, des tampons entre le bus et la fonction USB. En effet, il n'est pas possible pour le bus d'écrire directement dans la fonction, et pour la fonction d'écrire directement sur le bus. Les données sont donc stockées temporairement (jusqu'à ce que l'hôte ou le périphérique les lisent) dans les terminaisons. C'est donc pour cette raison que dans le paquet Jeton, l'hôte précise la terminaison à laquelle il veut s'adresser.

II.4.3.3. Les types de transferts

La spécification de l'USB définit 4 types de transferts entre l'hôte et le périphérique.

a) Les transferts de commande

Ce sont les transferts qui sont généralement utilisés pour les opérations de commande et d'état. L'énumération du périphérique par exemple, se fait en mode transfert de commande.
Ces transferts surviennent généralement en paquets directs et par rafales initiés par l'hôte, de manière à utiliser le meilleur rendement de livraison.
Le transfert de commande est fiable : en cas d'erreur sur un paquet, il est répété.

b) Les transferts d'interruption

Ce type de transfert est très utilisé, puisque c'est celui qui est mis en œuvre pour les souris, les claviers,… En fait, quand le périphérique a une donnée à transférer à l'hôte, il doit attendre que l'hôte l'interroge pour lui signaler qu'il a une information urgente à transférer. En fait, ce n'est pas réellement un système d'interruption au sens informatique du terme. L'hôte n'interrompt pas le transfert en cours avec un autre périphérique pour se précipiter vers le périphérique nécessitant un transfert urgent.

c) Les transferts isochrones

C'est certainement le mode de transfert le plus efficace en matière de débit, de disponibilité de la bande passante et du délai d'attente. Mais c'est le plus complexe. Il est utilisé principalement pour des données ayant des durées de vie critique telle que les trames audio ou vidéo. Ce type de transfert assure un débit minimum, mais il peut arriver que certains paquets soient erronés.

d) Les transferts en Bloc

Ce type de transfert est utilisé quand il faut transférer une grande quantité d'informations pendant un temps relativement court. Par exemple, un appareil photo ou un caméscope utilise ce type de transfert pendant lequel 90% de la bande passante du bus est attribué au périphérique et les paquets erronés sont répétés.

II.4.3.4. Les descripteurs USB

Tous les appareils USB ont une hiérarchie de descripteurs qui détaillent, pour le compte de l'hôte, des informations l'instruisant sur la nature de l'appareil, qui l'a réalisé, quelle version USB il supporte, de combien de manières il peut être configuré, le nombre de terminaisons et leurs types etc.
Les descripteurs les plus courants sont :

a) Descripteurs d'appareils

Le descripteur d'appareil représente l'appareil en entier. En conséquence un appareil USB ne peut avoir qu'un seul descripteur d'appareil. Il donne des informations élémentaires et pourtant fondamentales sur l'appareil telles la version USB supporté, la taille maximale de paquet, les IDs constructeurs et produits et le nombre de configurations possibles que peut avoir l'appareil.

b) Les Descripteurs d'Interfaces

Le Descripteur d'Interface peut être vu comme un " en tête " ou un regroupement de terminaisons dans un groupe fonctionnel exécutant une simple fonction pour l'appareil.

c) Les Descripteurs de Terminaisons

Les Descripteurs de Terminaisons sont utilisés pour décrire les terminaisons autres que la terminaison zéro. La terminaison zéro est toujours censée être une terminaison de commande et est configurée avant que n'importe quel autre descripteur ne soit sollicité. L'Hôte utilisera l'information renvoyée par ces descripteurs pour déterminer les besoins de bande passante du Bus.

d) Les Descripteurs de chaînes de caractères

Les Descripteurs de chaînes fournissent une information lisible pour l'homme et sont optionnels. S'ils ne sont pas utilisés, tout champ d'index de descripteurs de chaînes doit être mis à zéro indiquant qu'il n'y a pas de descripteur de chaîne disponible.

II.4.4. Les gammes USB

Même si la gamme des périphériques disponibles pour le bus USB est extrêmement vaste, on retrouve les classes d'applications suivantes :

- ✓ périphériques d'interaction avec l'utilisateur (HID) : claviers, souris, joystick.
- ✓ périphériques de stockage : disques durs externes, appareils photo, lecteurs multimédia, et surtout clés USB, un concept apparu spécifiquement pour le bus USB. Il s'agit de l'association d'une mémoire flash et d'une interface USB, le tout contenu dans un petit boîtier.

- ✓ multimédia et imagerie : imprimantes, scanners, cartes son, webcams, tuners TV, écran secondaire (intégrant son propre contrôleur vidéo) .
- ✓ adaptateurs de réseau ou de communication : Wifi, Ethernet, Bluetooth, infrarouge IrDA, Modem.
- ✓ Bus et interfaces : port série RS-232, port parallèle, port PS/2, port joystick, Bus CAN, GPIB (IEEE-488), port série RS-485.

II.4.4.1. La gamme choisie

La gamme qui nous intéresse est le périphérique **HID** « **Human Interface Device** ». Cette gamme regroupe les outils les plus utilisés comme les souris, claviers et joysticks. La vitesse de ces périphériques est limitée (64ko/s), vitesse largement suffisante pour notre application. Cette carte aura l'avantage de ne nécessiter aucun driver particulier. Windows possède en effet tous les pilotes nécessaires.

II.4.5. Programmation de la liaison USB

II.4.5.1. Programme PIC18F2550

La programmation de ce type de microcontrôleur se fait à l'aide des langages évolués (CAnsi), l'interprétation se fait à travers des compilateurs, tels que le C18 de Microchip, le CCS...
Pour pouvoir profiter du « Framework USB HID» distribué par Microchip on a choisi le C18, un compilateur complet qui garde la possibilité de manipuler les registres avec facilité, qui est un avantage très utile pour maitriser précisément les événements qui surviennent au cours du transfert de données.

Pour se faire, le Framework diffusé sur le site de Microchip a du subir une modification du point de vue du descripteur. Ce dernier est une partie du code source qui reprend l'usage et les différents paramètres de la carte USB, tels que son PID et VID qui permettent d'identifier le périphérique USB et d'ouvrir la communication, ainsi que toutes les données relatives au type d'application, au volume des paquets transmis, au débit maximum.
La procédure « ProcessIO » est exécutée cycliquement et nous permet de traiter les données reçues et d'en renvoyer de nouvelles.

Pour cela nous possédons quelques fonctions :

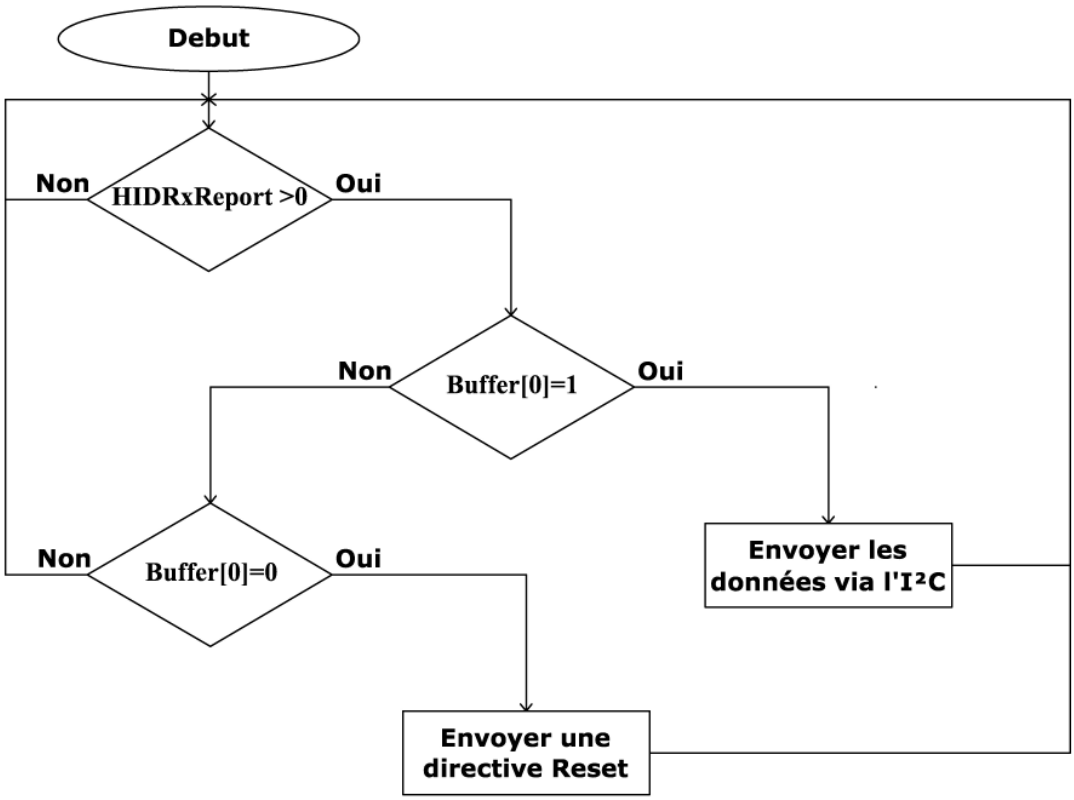
HIDRxReport(Buffer, PacketSize) : La fonction renverra une valeur supérieure à 0 si la carte a reçu un paquet dans « Buffer » de la taille de PacketSize.

mHIDTxIsBusy() : Permet de voir si la liaison USB est actuellement occupée (dans ce cas la fonction renvoie 1).

HIDTxReport(Buffer,PacketSize) : Permet de renvoyer un paquet contenu dans Buffer de la taille de PacketSize.

La vitesse de la communication ne dépend que de deux paramètres: la taille du paquet envoyé (maximum 64 octets) et la période séparant chaque demande du pic (« pooling » minimum 1 ms). Soit une réception par le pic de 64 octets toutes les 1 ms dans les meilleures conditions, c'est-à-dire 1000 paquets de 64 octets, donc 64 ko/s. Ces paramètres sont réglables dans le descripteur (« usbdsc.c »).

L'organigramme relatif à la réception d'une donnée sur le pic 18F2550 est donné de la manière suivante :



NB : Ce programme est exécuté cycliquement avec une condition sans fin.

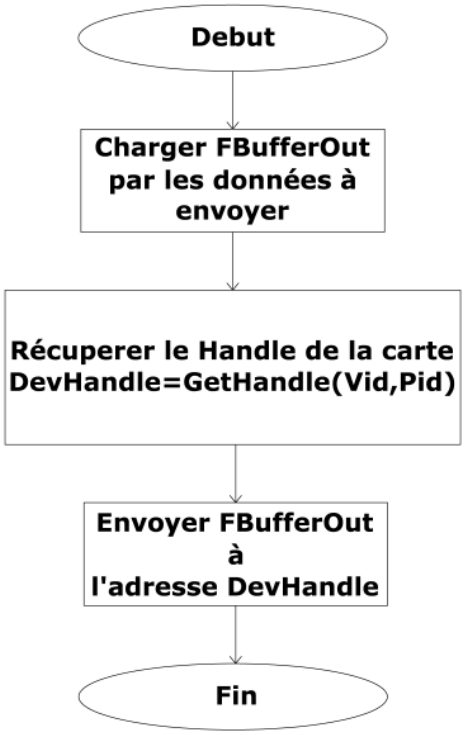
II.4.5.2. Programme PC

La communication USB nécessite la gestion d'un certain nombre d'événements et de procédures (ou fonctions). Pour se faire, on fait appel à une DLL fournie avec le logiciel EasyHID.

Dans le programme nous trouvons un tableau utilisable dans toute l'application de 64 éléments qui s'appelle « FBufferOut ». C'est dans ce tableau que l'on place les données à envoyer.

Une fois les données prêtes, nous devons créer la communication avec la carte. La première étape est d'interroger le programme sur l'adresse à laquelle il doit envoyer son paquet. Pour cela, « DevHandle » variable de type cardinal, se voit attribuer une valeur qui sera donnée par la fonction GetHandle (VendorID, ProductID). Il ne reste qu'à envoyer le tout par la fonction WriteUSB(DevHandle, @FBufferOut) .

L'organigramme relatif à l'envoi de données sur l'USB est donné de la manière suivante :



II.5. Le BUS I²C

II.5.1.Présentation

Le bus I²c est caractérisé par une liaison en mode série réalisée à l'aide de 2 fils. C'est la société Philips qui en a créé le concept au début des années 80.
Son succès est lié à sa simplicité.

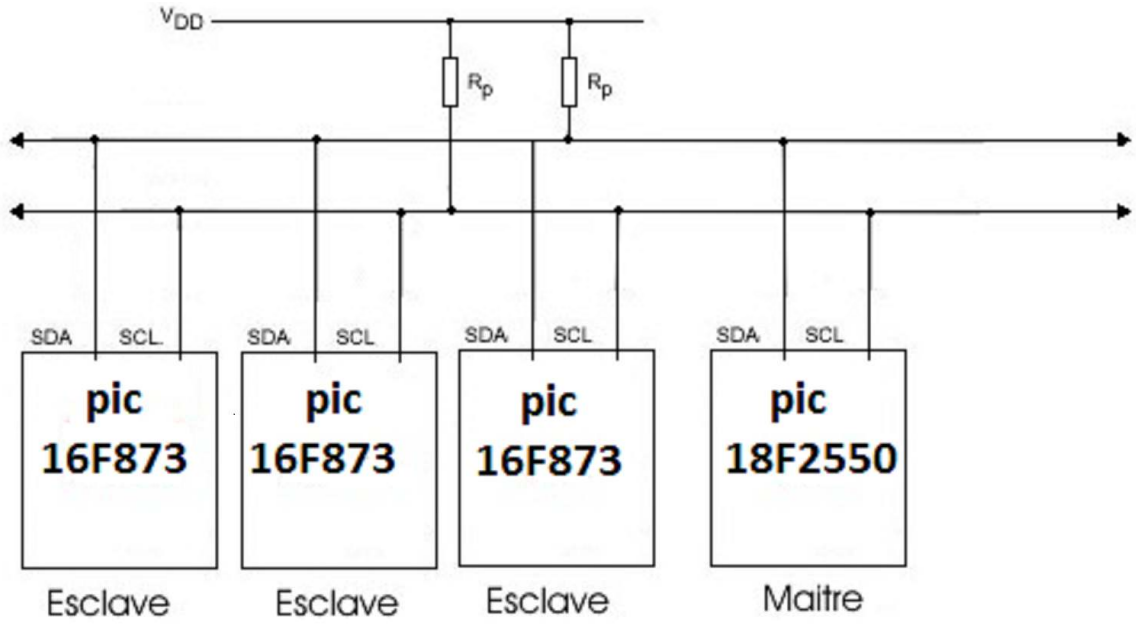


Fig. II.10 : Schéma synoptique de la mise en place d'une liaison I²C

Nous représentons ci-dessus l'architecture type d'un bus I²C, on remarque que plusieurs composants viennent se "greffer" sur le même bus. Les données transitent par les lignes :

- ✓ SDA : signal de donnée, généré par le Maitre ou l'Esclave.
- ✓ SCL : signal d'horloge généré par le Maitre.

La communication sur le bus est orchestrée de la manière suivante :
Le Maitre envoie sur le bus l'adresse du composant avec qui il doit communiquer. Chacun des esclaves a une adresse fixe. l'esclave qui se reconnaît répond à son tour par un signal de confirmation, puis le Maitre continue la procédure de communication.
Dans tout les cas, les transactions seront confirmées par un ACK.

II.5.2.Caractéristiques électriques

- ✓ Tension de bus : 5VDC.
- ✓ Fréquence maximale de fonctionnement : 400Khz (variable suivant les composants connectés au bus).
- ✓ Etat logique Haut : de 3 à 5 Volts.
- ✓ Etat logique Bas : de 0à 1.5 Volts.
- ✓ Capacité maximum admissible sur le bus : 400 Pf.
- ✓ Temps de montée des signaux : inférieur à 1 us.
- ✓ Temps de stabilité pour prise en compte d'un signal : supérieur à 5 us (règle générale).
- ✓ Résistance de rappel : à calculer en fonction de l'impédance du Bus (valeurs pratiques constatées de 1,5 K à 3,5 K).
- ✓ Distance de communication : Quelque dizaines de centimètres, et plusieurs mètres avec un tampon.

II.5.3. Les différents types de signaux

- ✓ La condition de repos :

Condition dans laquelle aucun composant ne communique, les lignes du bus sont à l'état Haut :

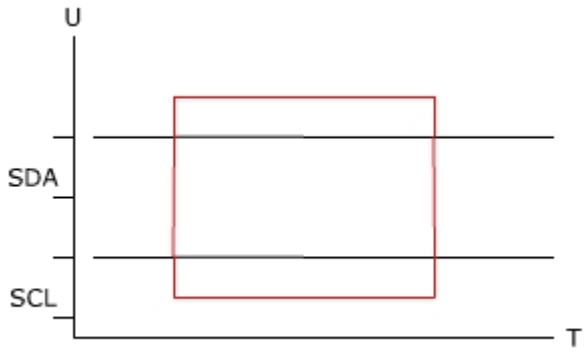


Fig. II.11 : Condition de repos.

✓ La condition de départ :

Le Maitre force la ligne SDA au niveau bas pour « réveiller » les Esclaves quand SCL est au niveau haut :

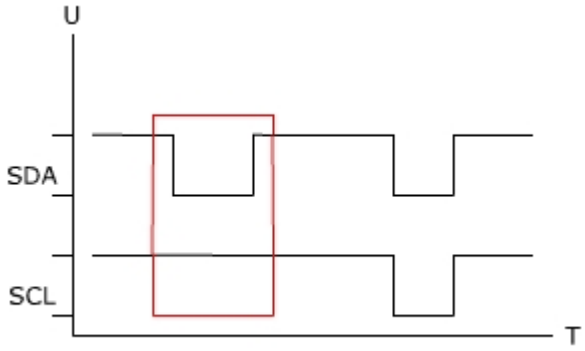


Fig. II.12 : Condition de départ.

✓ La condition d'arrêt :

Les lignes SDA et SCL sont au niveau bas, puis quand SCL passe au niveau haut, le Maitre libère la ligne SDA au niveau haut. A cet instant le bus est considéré comme disponible :

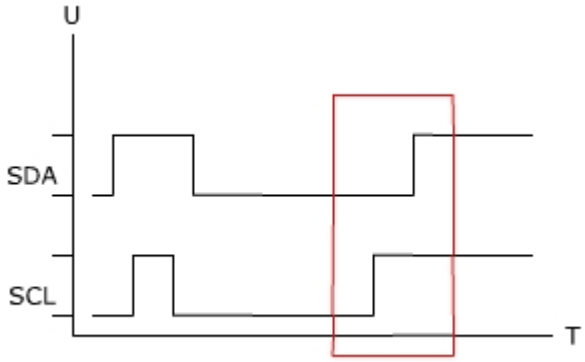


Fig. II.13 : Condition d'arrêt.

✓ L'acquittement :

Une fois les données transmises, un acquittement est opéré par celui qui reçoit les données. Cette procédure est réalisée en fin de trame de transmission, soit à la neuvième impulsion. Le récepteur maintient la ligne SDA au niveau bas pendant le front d'horloge :

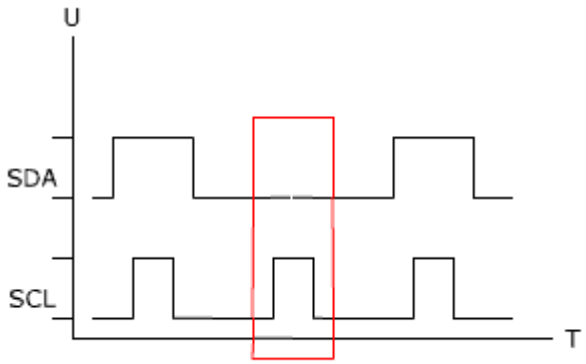


Fig. II.14 : Acquittement.

II.5.4. Trame de Communication

II.5.4.1.Architecture d'une trame

Nous avons vu précédemment les opérations de base effectuées sur le Bus I²C. Voyons maintenant comment assembler ces « morceaux » afin de constituer une trame complète.

Dans un premier temps, il faut réveiller le bus en réalisant une condition de départ, ensuite viennent s'intercaler les trames de données. L'Esclave ayant reçu les informations acquittera le Maitre pour lui faire savoir qu'il à bien reçu la trame de données, dans le cas contraire, c'est au Maitre de recommencer la transaction depuis le départ. Pour finir le signal de Stop mettra fin à toute communication.

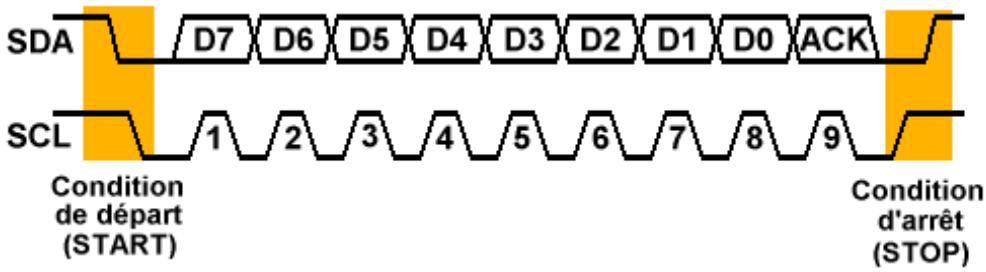


Fig. II.15 : Architecture d'une trame.

II.5.4.2.La transmission d'un octet

Avant de placer les bits qui forment l'octet à transmettre sur le bus, le Maître doit placer la ligne d'horloge SCL à 0. Tant que la ligne SCL est au niveau haut, la ligne SDA ne doit pas changer d'état, sinon cette condition sera interprétée comme la condition d'arrêt. La condition d'arrêt peut survenir même au milieu de la transmission d'un octet, pour abandonner la transmission et libérer le bus pour les autres circuits. Pour transmettre correctement les bits sur la ligne SDA, le Maître doit donc tout d'abord placer la ligne SCL à 0. Ensuite, le Maître peut placer la ligne SDA au niveau correspondant au bit à transmettre et replacer la ligne SCL au niveau 1 pour indiquer que le bit est présent sur la ligne SDA. La même opération va se répéter autant de fois que nécessaire pour transmettre les 8 bits de données.

Remarque : Le bit de poids fort est transmis en premier.

Une fois les 8 bits transmis, le circuit qui vient de recevoir les données doit imposer un bit d'acquittement ACK sur la ligne SDA. Pour cela, pendant que la ligne SCL est au niveau bas, le Maître place sa propre sortie au niveau haut, tandis que le récepteur (esclave) place sa sortie au niveau bas. Puisque les sorties sont à collecteur ouvert, la ligne SDA restera au niveau bas à cause de l'esclave. Le maître relit ensuite la ligne SDA une fois qu'il a passé la ligne SCL au niveau haut. Si la valeur lue pour le bit ACK est 0, c'est que l'esclave s'est bien acquitté de l'octet reçu, sinon c'est qu'il y a une erreur et le Maître doit générer la condition arrêt.

II.5.4.3.La transmission d'une adresse

Le nombre de composants qu'il est possible de connecter sur un bus I²C étant largement supérieur à deux, le maître doit pouvoir choisir quel esclave est censé recevoir les données. Dans ce but, le premier octet que transmet le maître n'est pas une donnée mais une adresse. Le format de l'octet d'adresse est un peu particulier puisque le bit D0 est réservé pour indiquer si le maître demande une lecture â l'esclave ou bien au contraire si le maître impose une écriture à l'esclave.

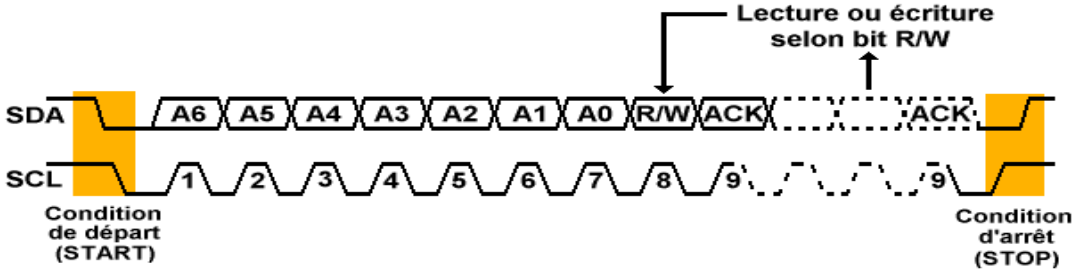


Fig. II.16 : Transmission d'une adresse.

Chaque circuit connecté au bus I²C possède une adresse, qui doit être unique. L'adresse associée à un composant est définie en partie par l'état de broches de sélections et d'autre part par sa fonction.

Une fois l'adresse envoyée sur le bus, l'esclave concerné doit répondre en plaçant le bit ACK à 0. Si le bit ACK vaut 1, le maître comprend qu'il y a une erreur de sélection et il génère la condition d'arrêt. En revanche, si le bit ACK vaut 0, le maître peut continuer les opérations.

II.5.4.4. Écriture d'une donnée

Si le bit R/W précédemment envoyé était à 0, cela signifie que le maître doit transmettre un ou plusieurs octets de données. Après chaque bit ACK valide, le maître peut continuer d'envoyer des octets à l'esclave ou bien il peut décider de terminer le dialogue par une condition d'arrêt.

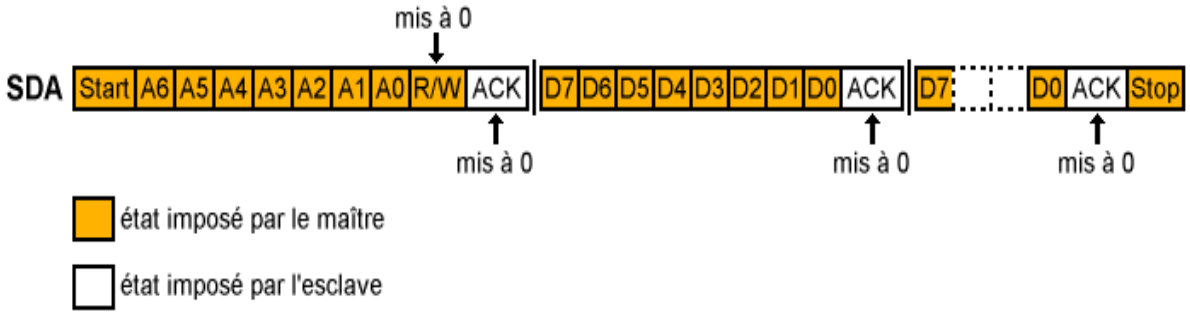


Fig. II.17 : Ecriture d'une donnée.

II.5.4.5. Lecture d'une donnée

Si le bit R/W transmis en même temps que l'adresse est à 1, cela signifie que le maître veut lire des données issues de l'esclave. C'est toujours le maître qui va générer le signal d'horloge SCL. En revanche, après le bit ACK de l'adresse, c'est l'esclave qui va garder le contrôle de la ligne SDA. Pour cela, le maître va placer sa propre sortie SDA au niveau haut pour permettre à l'esclave de prendre le contrôle de la ligne SDA. L'esclave doit alors scruter la ligne SCL et attendre le niveau bas pour changer l'état de la ligne SDA, faute de quoi le maître détectera une condition arrêt et abandonnera le transfert

Après que l'esclave ait transmis les 8 bits de données, c'est le maître, cette fois-ci, qui va générer un bit d'acquittement. Si le maître désire lire des octets supplémentaires, il placera le bit d'acquittement à 0. En revanche, si le maître décide que la lecture est terminée, il placera le bit ACK au niveau 1. L'esclave comprendra alors que le transfert est terminé. Cette fois-ci, bien que

le bit ACK soit au niveau 1, cela ne correspond pas à une condition d'erreur mais à une fin de transfert.

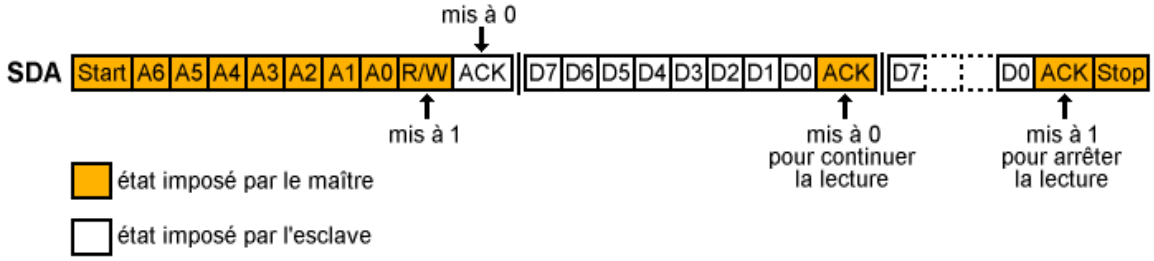


Fig. II.18 : Lecture d'une donnée.

II.5.4.6. Restart

Le protocole du bus I²C ne s'arrête pas là. Il est possible d'enchaîner écriture et lecture de l'esclave sans avoir à passer par une condition d'arrêt. Par exemple, dans le cas de la lecture d'une RAM, le maître commence par envoyer l'adresse du composant avec le bit R/W positionné sur l'écriture. La RAM adressé, en esclave renvoie ACK = 0. Ensuite, le maître transmet l'adresse interne de la case mémoire demandée. Une fois encore, l'esclave répond par ACK = 0. Le maître envoie alors à nouveau une condition de départ (sans passer par une condition d'arrêt), puis de nouveau l'adresse du composant sélectionnée mais en plaçant le bit R/W sur la position lecture. L'esclave va répondre par ACK = 0 et enchaîner par la transmission du contenu de la case mémoire demandée. C'est toujours le maître qui impose l'horloge SCL mais c'est l'esclave, en l'occurrence la RAM, qui contrôle la ligne SDA. Une fois les 8 bits de données transmis par la RAM, si le maître veut lire le contenu de la case mémoire suivante, il placera le bit ACK au niveau 0. Dans ce cas, la RAM recommence la lecture avec la case mémoire suivante. En revanche, si le maître souhaite en terminer avec la lecture, il placera le bit ACK au niveau 1 et il générera ensuite la condition d'arrêt.

Le contenu des octets de données lus ou écrits aura une signification qui dépend du composant sélectionné. Mais le protocole reste le même.

II.5.4.7. Les adresses réservées

Les adresses 0000 0xxx ne sont pas utilisées pour l'adressage de composants. Elles ont été réservées par Philips pour effectuer certaines fonctions spéciales.

Adresse d'appel général : 0000 0000

Après l'émission d'un appel général, les circuits ayant la capacité de traiter ce genre de demande d'appel émettent un acquittement.

Le deuxième octet permet de définir le contenu de l'appel :

0000 0110 : RESET Remet tous les registres de circuits connectés dans leur état initial (mise sous tension). Les circuits qui le peuvent rechargent leur adresse esclave.

0000 0100 : Les circuits définissant leur adresse de façon matérielle réinitialisent leur adresse esclave. Cela ne réinitialise pas les circuits.

0000 0000 Interdit.

xxxx xxx1 Cette commande joue le rôle d'interruption. xxxx xxx peut être l'adresse du circuit qui a généré l'interruption.

Les autres valeurs du second octet ne sont pas définies et sont tout simplement ignorées.

Octet de start : 0000 0001 Cet octet est utilisé pour synchroniser les périphériques lents avec les périphériques rapides.

Début d'adressage CBUS : 0000 001x
L'émission de cet octet permet de rendre sourd tous les circuits I²C présents sur le bus. À partir de ce moment, on peut transmettre ce que l'on désire sur le bus, en utilisant par exemple un autre protocole. Le bus repasse en mode normal lors de la réception d'une condition d'arrêt.

Autre : 0000 0110 à 0000 1111
Ces adresses ne sont pas définies et sont ignorés par les circuits I²C. Elles peuvent être utilisées pour débugger un réseau multi master.

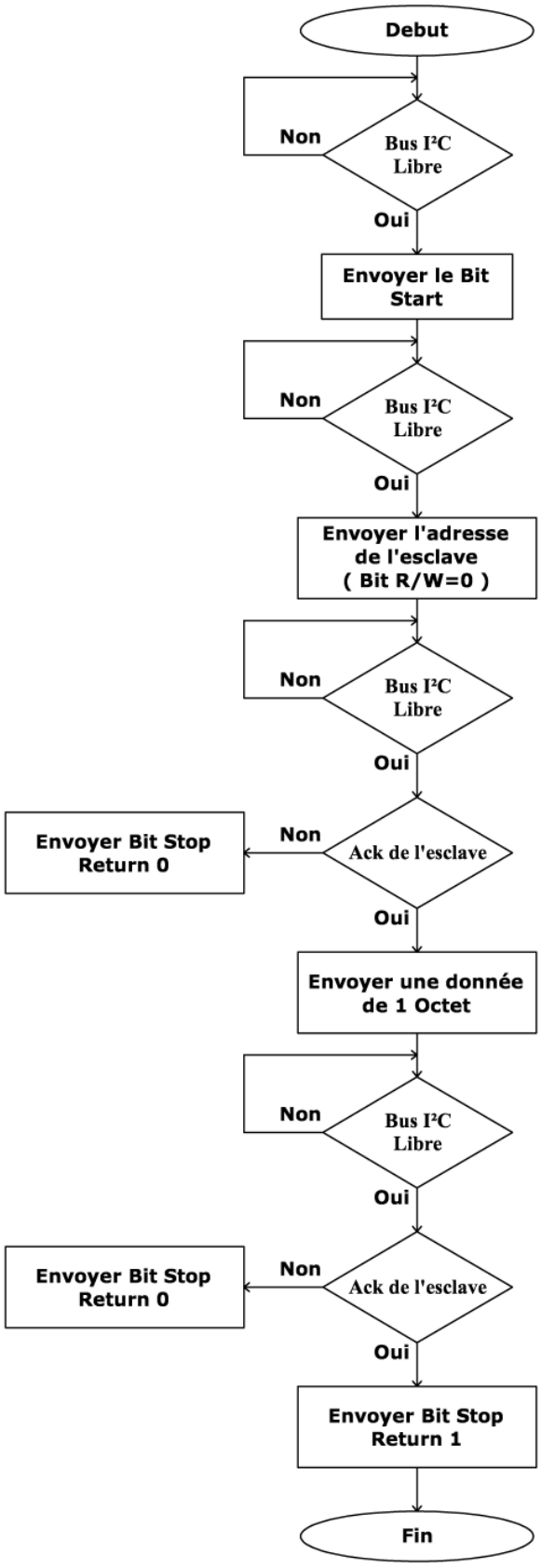
II.5.5. Programmation de la communication I²C

II.5.5.1. Programmation du PIC18F2550 (Maitre)

Une fois la donnée reçue, elle est accessible depuis le Buffer qui contient les informations destinées aux trois moteurs. Le PIC18F2550 joue le rôle d'intermédiaire entre le PC et les trois Pics moteurs de telle manière à envoyer chaque information au PIC moteur approprié.

Pour cela on a développé une fonction « envoyer_i2c» qui reçoit comme paramètres l'adresse et la donnée du PIC de destination et procède ainsi à l'envoi sur le Bus I²C.

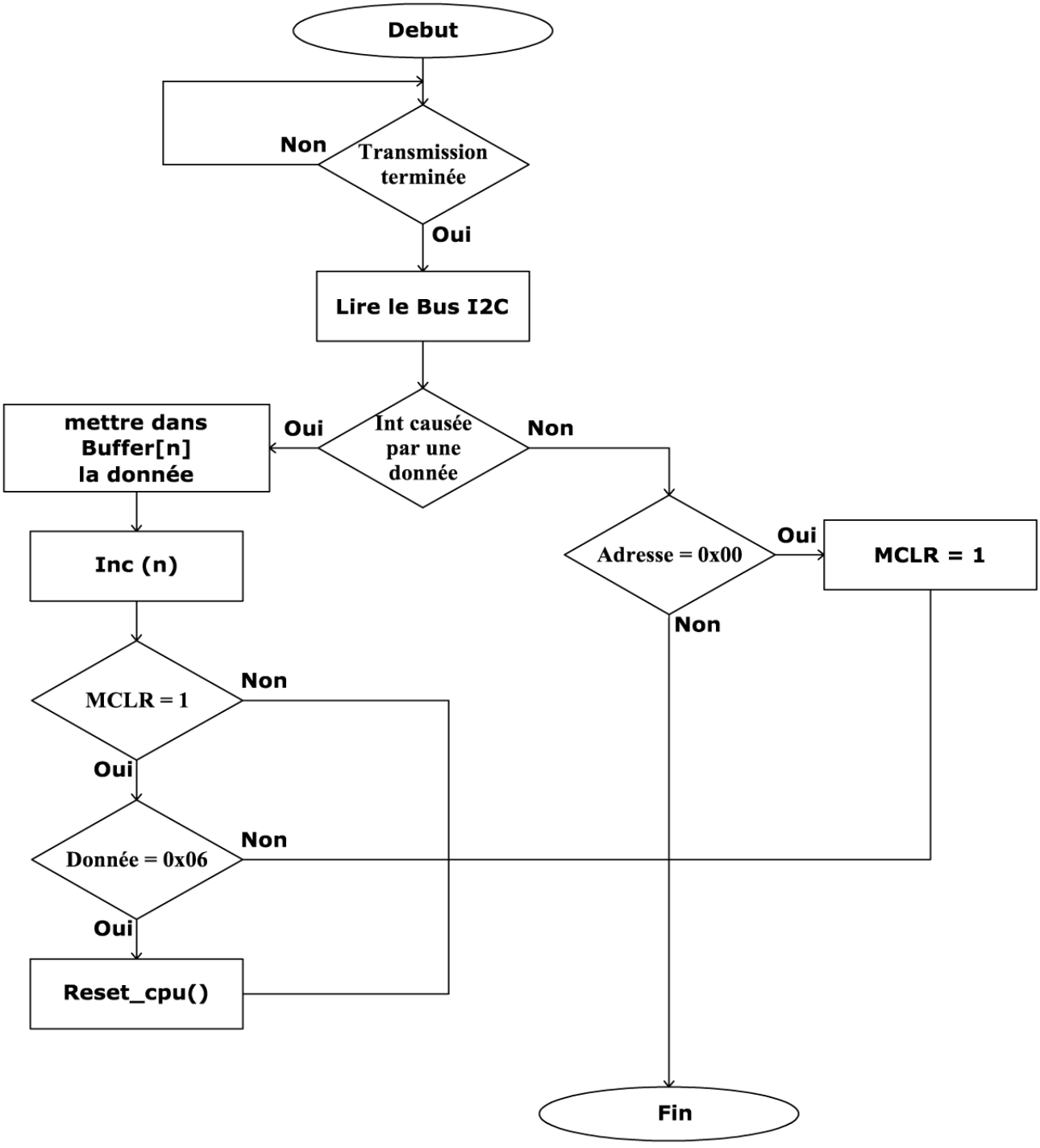
L'organigramme relatif à la fonction « envoyer_i2c » est donné de la manière suivante :



II.5.5.2. Programmation du PIC16F873A (Esclave)

Pour laisser place au programme qui génère le signale permettant la rotation du moteur (dit exécutif), on a procédé à la gestion d'acquisition de données sur le Bus I²C par interruptions.
Les données s'acheminent octet par octet vers un Buffer prévu pour leurs recueils.

L'organigramme relatif à la procédure de gestion de l'interruption I²C est donné de la manière suivante :



II.6. Carte de commande moteur

II.6.1. Principe de fonctionnement

Ce bloc devra faire l'interprétation et la conversion d'une information reçue à son entrée, en un ensemble de signaux de puissance à sa sortie, converties à leurs tours en mouvement par le moteur.

Grâce à l'emploi des circuits intégrés spécialisés (L297, L298), l'utilisateur n'aura pas à se soucier de la construction des séquences. Il suffira en effet de disposer des signaux suivants :

- ✓ un signal d'horloge permettant l'avance d'un pas
- ✓ Un niveau logique déterminant le sens de rotation
- ✓ Un niveau logique fixant l'amplitude des pas : pas entier ou demi-pas
- ✓ Une impulsion de remise à zéro
- ✓ Un niveau logique validant le circuit de commande

Une fois les niveaux logiques déterminés, il ne sera plus nécessaire que d'envoyer des impulsions afin d'obtenir la rotation de l'axe du moteur.

Le schéma de principe est donné en fig. II.19 :

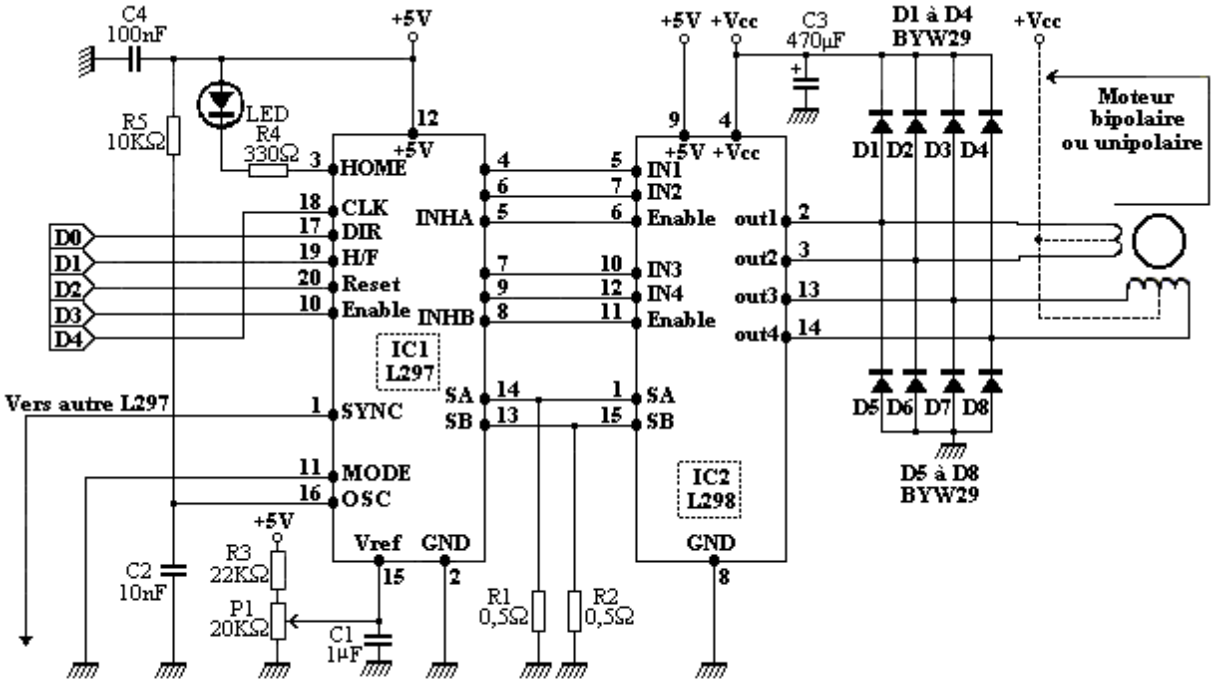


Fig. II.19 : Schéma électrique de la carte de commande du moteur.

❖ **Nomenclature :**

Résistances :
- ✓ R_1, R_2 : 0,5 Ω (ou 0,47 Ω) 3W
- ✓ R_3 : 22 KΩ
- ✓ R_4 : 330 Ω
- ✓ R_5 : 5K Ω
- ✓ P_1 : ajustable multi tours 20 KΩ

Condensateurs :
- ✓ C_1 : 1µF
- ✓ C_2 : 10 nF
- ✓ C_3 : 470 µF
- ✓ C_4 : 100 nF

Semi-conducteurs :
- ✓ D_1 à D_8 : Diodes rapides BYW29 ou BY251
- ✓ DEL_1 : diode Led verte

Circuits intégrés :
- ✓ IC_1 : L297
- ✓ IC_2 : L298

Divers
- ✓ 1 dissipateur thermique (L298)
- ✓ 1 support pour IC_1 20 broches
- ✓ 2 borniers à vis à 2 points
- ✓ 1 bornier à vis à 4 points

II.6.2. Fonctions des broches du L297

La platine est commandée à partir des entrées de données (D_0 à D_4) :
- ✓ D_0 (broche 17 : DIR) fixe le sens de rotation.
- ✓ D_1 (broche 19 : H/F) détermine le mode d'avance en pas entiers ou en demi-pas.
- ✓ D_2 (broche 20 : /RESET) provoque la remise à zéro.
- ✓ D_3 (broche 10 : ENABLE) valide le fonctionnement du circuit.
- ✓ D_4 (broche 18 : /CLK) permet la validation d'un pas (ou d'un demi-pas).
- ✓ La broche 15, entrée de la tension de référence du circuit (Vref), est reliée à un pont diviseur constitué de deux résistances. L'une est fixe (R_3) et l'autre, ajustable (P_1) qui permet de fixer la valeur du courant débité par les étages de puissance du circuit intégré L298.
- ✓ Les résistances R_1 et R_2 d'une valeur de 0,5 Ω, mesurent ce courant et les entrées SA (broche 14) et SB (broche 13) sont les entrées de mesure, qui sont reliées également aux entrées du L298 (broches 1 et 15).
- ✓ Une LED munie de sa résistance (R_4) de limitation de courant est branchée sur la sortie HOME (broche 3). Elle signale le passage par la position initiale (ABCD = 0101) (indique le moment où un cycle de rotation a été accompli).
- ✓ La broche MODE (broche 11) est connectée à la masse pour permettre une décharge plus rapide aux bobinages du moteur.

- ✓ Le réseau RC constitué de la résistance (R_5) et du condensateur (C_2), relié à la broche 16, détermine la fréquence de fonctionnement de l'oscillateur de découpage interne.
- ✓ La broche SYNC (broche 1) est prévue dans le cas d'utilisation d'un autre L297 en parallèle, dans ce cas les deux broches sont reliées entre elle pour la synchronisation, et la broche OSC du second circuit intégré sera raccordée à la masse.
- ✓ La broche 2 est la connexion de la masse du circuit.
- ✓ La broche 12 est l'alimentation +5V du circuit de commande.
- ✓ Les sorties de commande A et B (broches 6 et 4) et C et D (broches 7 et 9), ainsi les broches INHA (5) et INHB (8) du L297, sont connectées au entrées du circuit de puissance (le L298).

II.6.3. Fonctions des broches du L298

- ✓ Le L298 devra être alimenté par deux tensions distinctes : le +5V (broche 9) pour toute la logique et le +Vcc (broche 4) moteur qui sera choisi en fonction de celui-ci. La valeur maximale de cette tension est de 46V (pour notre cas le Vcc = 12V).
- ✓ La connexion de la masse au circuit se fait sur la broche 5 (GND).
- ✓ Les sorties (broches 2, 3, 13 et 14) des étages de puissance du L298 sont directement connectées aux bobinages du moteur pas à pas. Quatre paires de diodes sont utilisées afin d'assurer une protection pour les transistors des ponts de puissance.

II.6.4. Modes de commande des moteurs

II.6.4.1. La commande en mode biphasé

Le mode biphasé est sélectionné par l'envoi d'un niveau bas sur l'entrée HALF/FULL du L297, quand le séquenceur est à un état impair. Deux phases sont alimentées à la fois.

Pour garantir le fonctionnement en biphasé, on fait un reset (envoi d'une impulsion sur l'entrée reset) pour remettre le séquenceur à l'état HOME (case 1). Voir (Fig. II.20) :

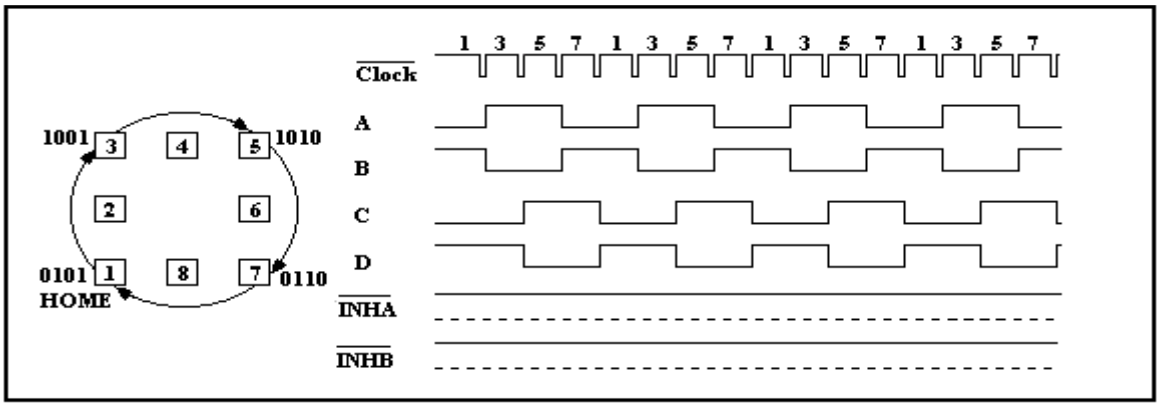


Fig. II.20 : Mode biphasé

II.6.4.2. La commande en mode monophasé

Le mode monophasé est sélectionné également par l'envoi d'un niveau bas sur l'entrée HALF/FULL du L297, mais cette fois-ci c'est quand le séquenceur est à un état pair. Une seule phase du moteur est alimentée à la fois.

Pour garantir le fonctionnement en mode monophasé, il faut d'abord remettre le séquenceur à l'état HOME (par l'envoi d'une impulsion sur l'entrée reset). Pour atteindre la case 2 (l'état 0001) le séquenceur doit avancer d'un demi-pas, sans sollicitation du moteur, en mettant la broche Enable à zéro pendant l'avance du demi-pas. Ensuite on doit remettre l'Enable à 1 pour le fonctionnement en mode monophasé. Voir (Fig. II.21) :

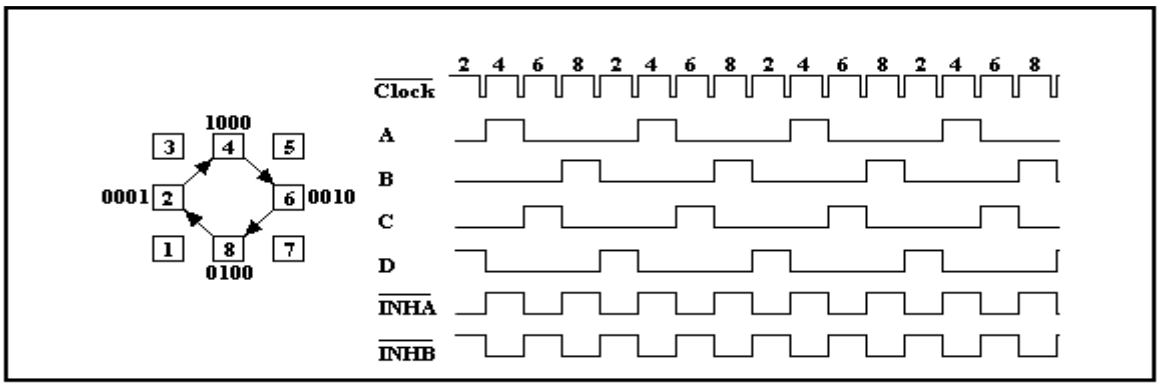


Fig. II.21 : Mode monophasé

II.6.4.3. La commande en mode demi-pas

Le mode demi-pas est sélectionné par l'envoi d'un niveau haut sur l'entrée HALF/FULL du L297. Alternativement une phase puis deux phases sont alimentées. Voir (Fig. II.22) :

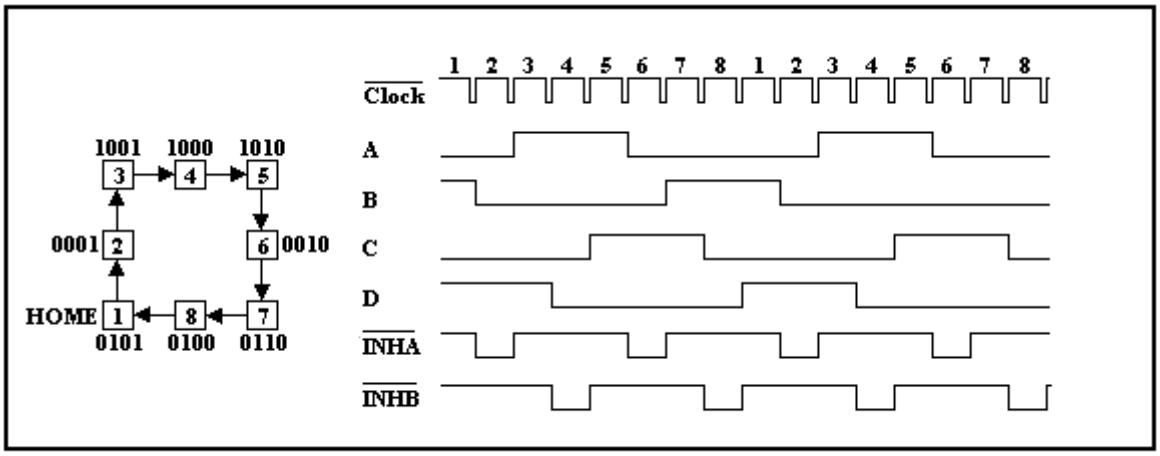


Fig. II.22 : Mode demi-pas.

II.6.5. Le mode choisi

Dans notre cas, la commande en mode biphasé a été choisie, pour profiter de la puissance maximale (couple élevé) car deux phases sont alimentées au même temps, contrairement au mode monophasé.

II.7 L'alimentation utilisée :

L'intensité du courant maximale que consomme un bobinage du moteur est de 2A, cela dit le moteur consomme 4A (2x2A). Ayant trois moteurs à alimenter, ça nous fait un courant maximal de 12A (3x4A), donc une alimentation ATX qui peut délivrer jusqu'à 20A sous 12V est suffisante pour alimenter non seulement les trois moteurs mais aussi tout le système (en 5V).

III.1. Introduction

Cette section donne une brève description de l'interpréteur de commandes et ses interactions avec les organes de notre centre d'usinage.

Le langage RS274/NGC (G-Code) et les fonctions d'usinage standards qui en découlent permettent d'envisager un centre d'usinage sous deux points de vue:

- ✓ Les composants mécaniques de la machine.
- ✓ Les composants de contrôle et les données utilisées pour le contrôle de la machine.

III.2. Le langage RS274/NGC

III.2.1. Vue d'ensemble du langage

Le langage RS274/NGC est basé sur des lignes de code. Chaque ligne (également appelée un "block") peut inclure des commandes pour faire produire diverses actions au centre d'usinage. Plusieurs lignes de code peuvent être regroupées dans un fichier pour créer un programme.

Une ligne de code typique commence par un numéro de ligne optionnel suivi par un ou plusieurs "mots". Un mot commence par une lettre suivie d'un nombre (ou quelque chose qui permet d'évaluer un nombre). Un mot peut, soit donner une commande, soit fournir un argument à une commande. Par exemple, "G1 X3" est une ligne de code valide avec deux mots. "G1" est une commande qui signifie "déplaces toi en ligne droite à la vitesse programmée" et "X3" fournit la valeur d'argument (la valeur de X doit être 3 à la fin du mouvement). La plupart des commandes RS274/NGC commencent avec G ou M (G pour Général et M pour Miscellaneous (auxiliaire)). Les termes pour ces commandes sont "G codes" et "M codes."

Le langage RS274/NGC n'a pas d'indicateur de début et de fin de programme. L'interpréteur cependant traite les fichiers. Un programme simple peut être en un seul fichier, mais il peut aussi être partagé sur plusieurs fichiers. Un fichier peut être délimité par le signe pourcent de la manière suivante. La première ligne non vide d'un fichier peut contenir un signe "%" seul, éventuellement encadré d'espaces blancs, ensuite, à la fin du fichier on doit trouver une ligne similaire. Délimiter un fichier avec des % est facultatif si le fichier comporte un M2 ou un M30, mais est requis sinon. Une erreur sera signalée si un fichier a une ligne pourcent au début, mais pas à la fin. Le contenu utile d'un fichier délimité par pourcent s'arrête après la seconde ligne pourcent. Tout le reste est ignoré.

Le langage RS274/NGC prévoit les deux commandes (M2 ou M30) pour finir un programme. Le programme peut se terminer avant la fin du fichier. Les lignes placées après la fin d'un programme ne seront pas exécutées. L'interpréteur ne les lit même pas.

III.2.2. Format d'une ligne

Une ligne de code permise par la norme RS274/NGC est construite de la façon suivante, dans l'ordre avec la restriction à un maximum de 256 caractères sur la même ligne.

1. Un block optionnel contenant le caractère d'effacement de ligne, barre de fraction "/".
2. Un numéro de ligne optionnel.
3. N'importe quel nombre de mots, valeurs de paramètres et commentaires.
4. Un caractère de fin de ligne (retour chariot ou saut de ligne ou les deux).

NB:

- ✓ Toute entrée non explicitement permise est illégale, elle provoquera un message d'erreur de l'interpréteur.
- ✓ Les espaces sont permis ainsi que les tabulations dans une ligne de code dont ils ne changent pas la signification, excepté dans les commentaires. Ceci peut donner d'étranges lignes, mais elles sont autorisées. La ligne " g0x +0. 12 34y 7 " est équivalente à "g0 x+0.1234 y7", par exemple.
- ✓ Les lignes vides sont permises, elles seront ignorées.
- ✓ Toutes les lettres en dehors des commentaires peuvent être, indifféremment des majuscules ou des minuscules sans changer la signification de la ligne.

III.2.3. Commandes et modes machine

En RS274/NGC, de nombreuses commandes produisent, d'un mode à un autre, quelque chose de différent au niveau de la machine, le mode reste actif jusqu'à ce qu'une autre commande ne le révoque, implicitement ou explicitement. Ces commandes sont appelées "modales". Par exemple, une commande G1 (déplacement linéaire) se trouve sur une ligne, elle peut être utilisée sur la ligne suivante avec seulement un mot d'axe, tant qu'une commande explicite est donnée sur la ligne suivante en utilisant des axes ou un arrêt de mouvement.

Les codes "non modaux" n'ont d'effet que sur la ligne ou ils se présentent. Par exemple, G4 (tempo) est non modale.

III.2.4. Groupes modaux

Les commandes modales sont arrangées par lots appelés "groupes modaux", à tout moment, un seul membre d'un groupe modal peut être actif. En général, un groupe modal contient des commandes pour lesquelles il est logiquement impossible que deux membres soient actifs simultanément, comme les unités en pouces et les unités en millimètres. Un centre d'usinage peut être dans plusieurs modes simultanément, si seulement un mode pour chaque groupe est actif. Les groupes modaux sont visibles dans le tableau suivant :

Signification du groupe modal	Mots des membres
Mouvements ("Groupe 1")	G0 G1 G2 G3 G33 G38.x G73 G80 G81 G82 G83 G84 G85 G86 G87 G88 G89
Choix du plan de travail	G17 G18 G19
Mode diamètre/rayon sur les tours	G7, G8
Mode de déplacements	G90, G91
Mode de vitesses	G93, G94
Unités machine	G20, G21
Correcteurs de rayon d'outil	G40, G41, G42, G41.1, G42.1
Correcteurs de longueur d'outil	G43, G43.1, G49
Options plan de retrait	G98, G99
Systèmes de coordonnées	G54, G55, G56, G57, G58, G59, G59.1, G59.2, G59.3
Types d'arrêt de programme	M0, M1, M2, M30, M60
Appel d'outil	M6
Types de rotation de la broche	M3, M4, M5
Arrosages	M7, M8, M9. Cas spéciaux: M7, M8 peuvent être actifs simultanément
Boutons de correction de vitesse	M48, M49
Contrôle de flux	O-
Codes non modaux ("Groupe 0")	G4, G10, G28, G28.1, G30, G30.1, G53 G92, G92.1, G92.2, G92.3 M100 à M199

Tab. III.1: Table des Groupes modaux.

III.2.5. Ordre d'exécution

L'ordre d'exécution des éléments d'une ligne est essentiel à la sécurité et l'efficacité d'une machine. Les éléments sont exécutés dans l'ordre indiqué ci-dessous, s'ils se trouvent sur la même ligne.

1. Choix de l'unité de longueur (G20, G21).
2. Réglage du mode de déplacement (G90, G91).
3. Réglage du type de retrait (G98, G99).
4. Effectuer les mouvements (G0 à G3, G33, G80 à G89).
5. Arrêt (M0, M1, M2, M30, M60).

III.3. Conception de l'interpréteur de commandes

Une fois que le programme G-Code est généré par un logiciel spécifique (Mastercam), notre interpréteur de commandes (développé en Borland Delphi) transforme les lignes de commandes en instructions qui ordonnancent le mouvement des trois axes du centre d'usinage.

Les instructions retournées par l'interpréteur sont destinées à informer le pic_moteur du nombre de pas, de la fréquence et du Sens de rotation des moteurs.

III.3.1. Les cycles gérés par l'interpréteur de commandes

- ✓ **G0: Interpolation linéaire en vitesse rapide :** Pour un mouvement linéaire en vitesse rapide, programmer G0 axes, au moins un mot d'axe doit être présent, les autres sont optionnels. Le G0 est optionnel si le mode mouvement courant est déjà G0. Cela produira un mouvement linéaire vers le point de destination à la vitesse rapide. Il n'est pas prévu d'usiner la matière quand une commande G0 est exécutée.

- ✓ **G1: Interpolation linéaire en vitesse travail :** Pour un mouvement linéaire en vitesse travail, programmer G1 axes, au moins un mot d'axe doit être présent, les autres sont optionnels. Le G1 est optionnel si le mode mouvement courant est déjà G1. Cela produira un mouvement linéaire vers le point de destination à la vitesse de travail.

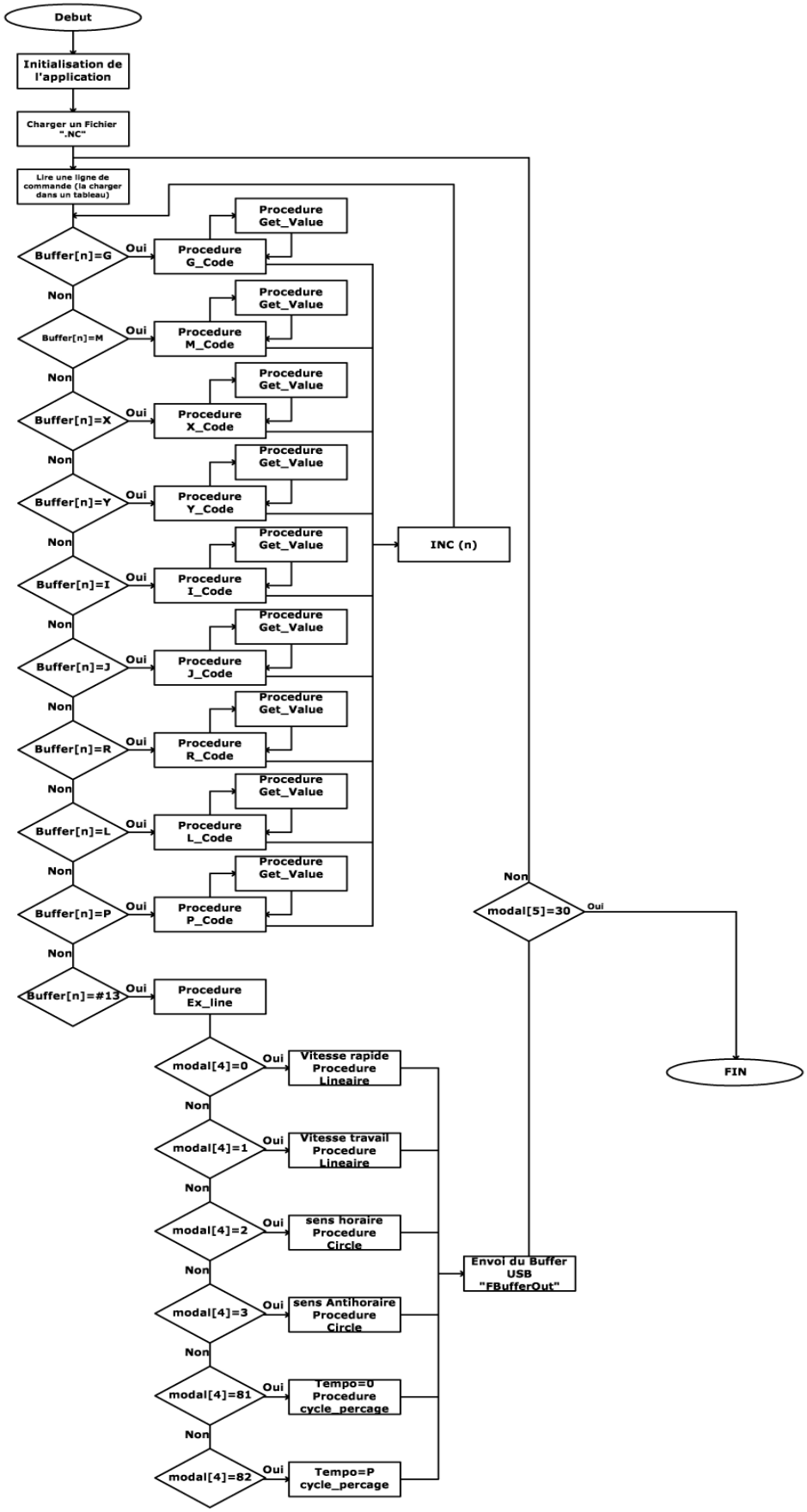
- ✓ **G2, G3: Interpolation circulaire en vitesse travail :** Un mouvement circulaire ou hélicoïdal est spécifié en sens horaire avec G2 ou en sens antihoraire avec G3. Les axes du cercle ou de l'hélicoïde, doivent être parallèles à l'axe Z. Si l'arc est circulaire, il se trouve dans un plan parallèle au plan (X, Y).

- ✓ **G20, G21: Choix des unités machine :** Programmer G20 pour utiliser le pouce comme unité de longueur. Programmer G21 pour utiliser le millimètre.

- ✓ **G81: Cycle de perçage :** Le cycle G81 est destiné au perçage. Programmer G81 X- Y- Z- R- L- donnera:
  1. Un mouvement préliminaire.
  2. Un déplacement de l'axe Z seul à la vitesse programmée, vers la position Z programmée.
  3. Retrait de l'axe Z en vitesse rapide jusqu'au plan de retrait R.

- ✓ **G82: Cycle de perçage avec temporisation :** Le cycle G82 est destiné au perçage. Programmer G82 X- Y- Z- R- L- P- donnera :
  1. Un mouvement préliminaire.
  2. Un déplacement de l'axe Z seul en vitesse programmée, vers la position Z programmée.
  3. Une temporisation de P secondes.
  4. Retrait de l'axe Z en vitesse rapide jusqu'au plan de retrait R.

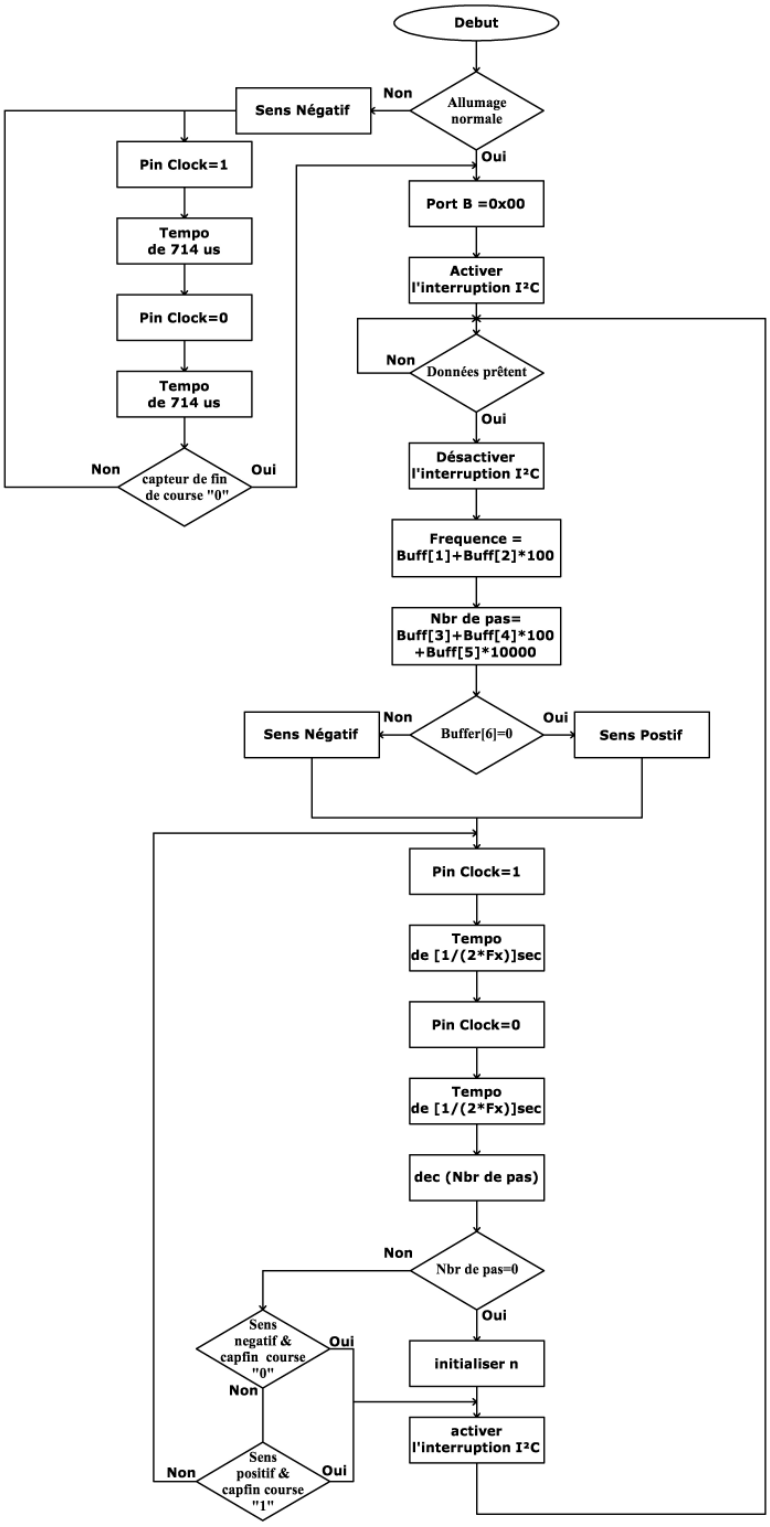
L'organigramme relatif est donné de la manière suivante :


Debut
Initialisation de l'application
Charger un Fichier ".NC"
Lire une ligne de commande (la charger dans un tableau)
Buffer[n]=G
Oui
Procedure G_Code
Non
Buffer[n]=M
Procedure M_Code
Buffer[n]=X
Procedure X_Code
Buffer[n]=Y
Procedure Y_Code
Buffer[n]=I
Procedure I_Code
Buffer[n]=J
Procedure J_Code
Buffer[n]=R
Procedure R_Code
Buffer[n]=L
Procedure L_Code
Buffer[n]=P
Procedure P_Code
Buffer[n]=#13
Procedure Ex_line
Procedure Get_Value
INC (n)
modal[5]=30
FIN
modal[4]=0
Vitesse rapide Procedure Lineaire
modal[4]=1
Vitesse travail Procedure Lineaire
modal[4]=2
sens horaire Procedure Circle
modal[4]=3
sens Antihoraire Procedure Circle
modal[4]=81
Tempo=0 Procedure cycle_percage
modal[4]=82
Tempo=P cycle_percage
Envoi du Buffer USB "FBufferOut"

III.4. Programme exécutif du Pic 16F873A

A ce stade toutes les informations nécessaires pour exécuter le mouvement des axes sont prêtes, il reste à développer une routine qui génère un signale carré et procède à la configuration des cinq entrées mis à part le clock de la carte de commande moteur.

L'organigramme correspondant est donné de la manière suivante :



IV.1. Réalisations pratiques

IV.1.1. Réalisation mécanique

Les photos ci après donnent des vue, sous différents angles de la structure mécanique de notre machine.

Fig. IV.1 : Vue de face de la fraiseuse.

Fig. IV.2: Vue de dessue de la fraiseuse.

IV.1.2. Réalisation électronique

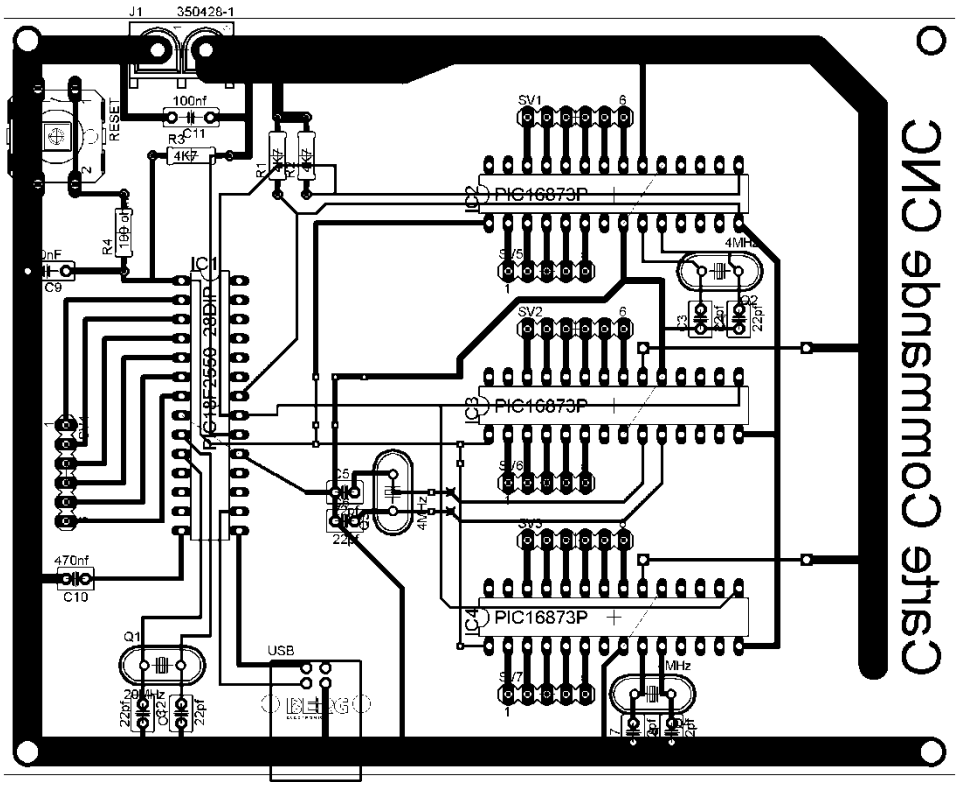


Fig. IV.3: Implémentation des composants de la carte de commande principale.

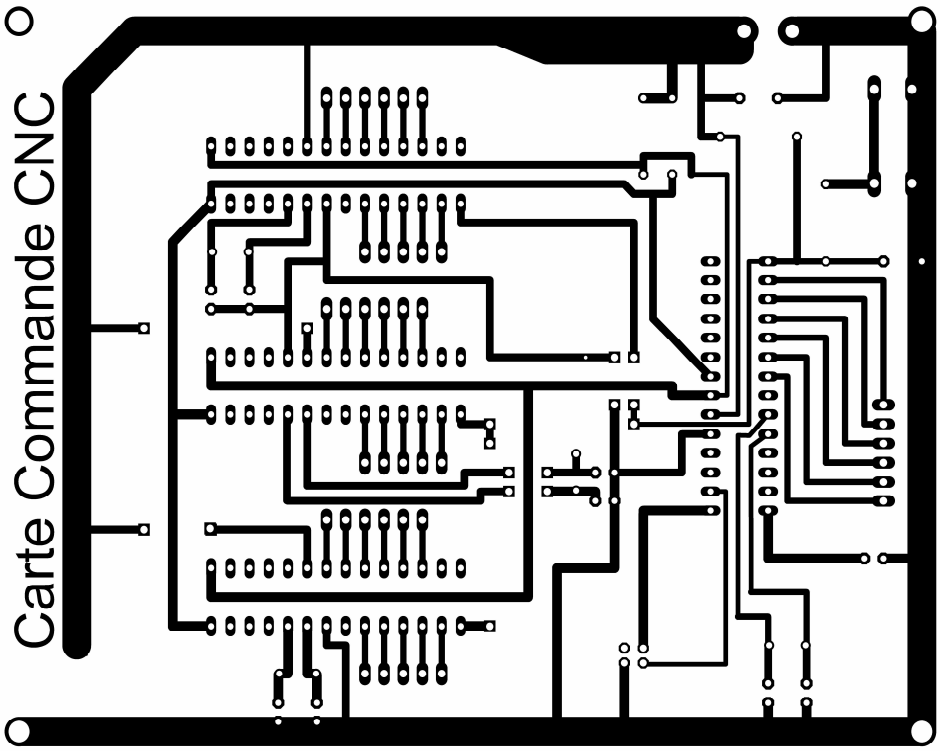


Fig. IV.4: Circuit imprimé de la carte de commande principale.

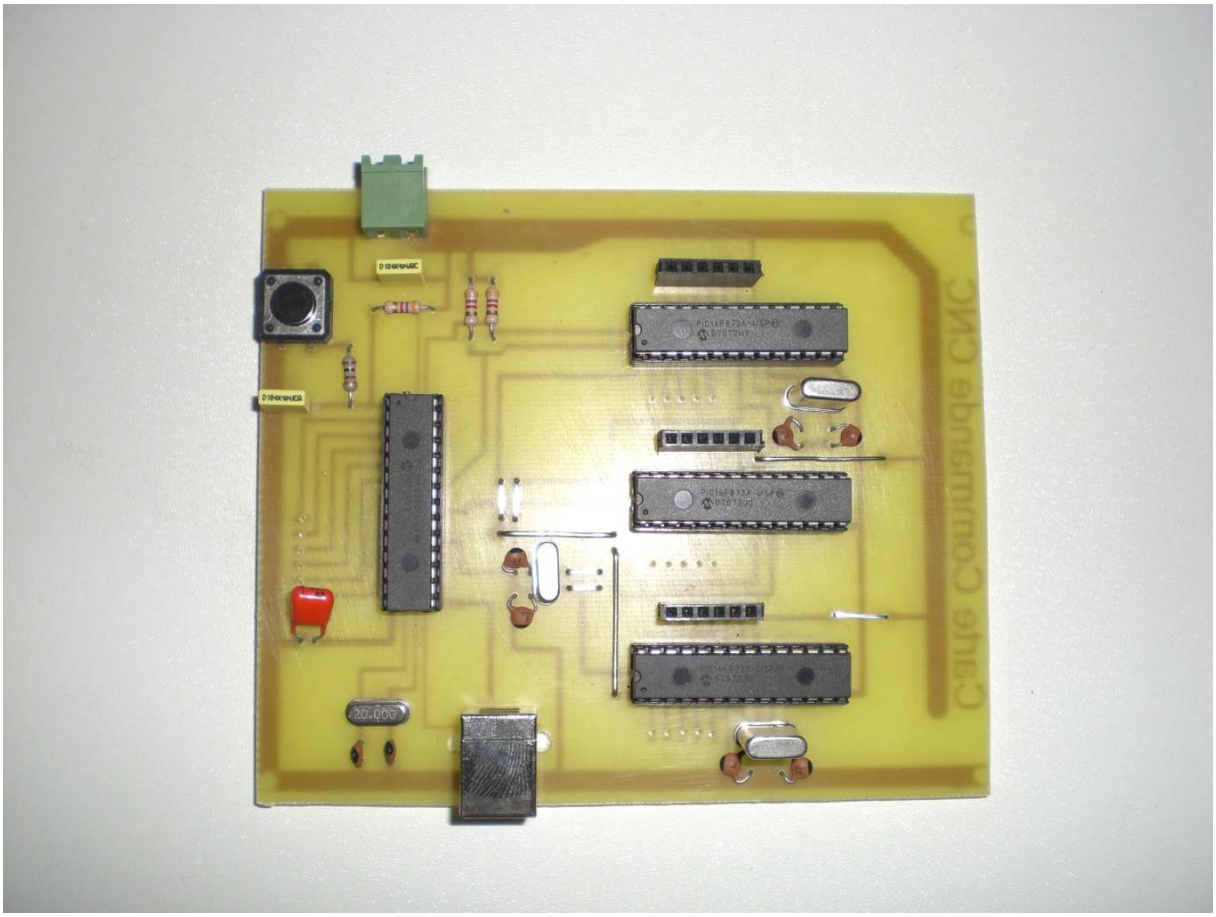


Fig. IV.5: Vue sur la carte de commande principale.

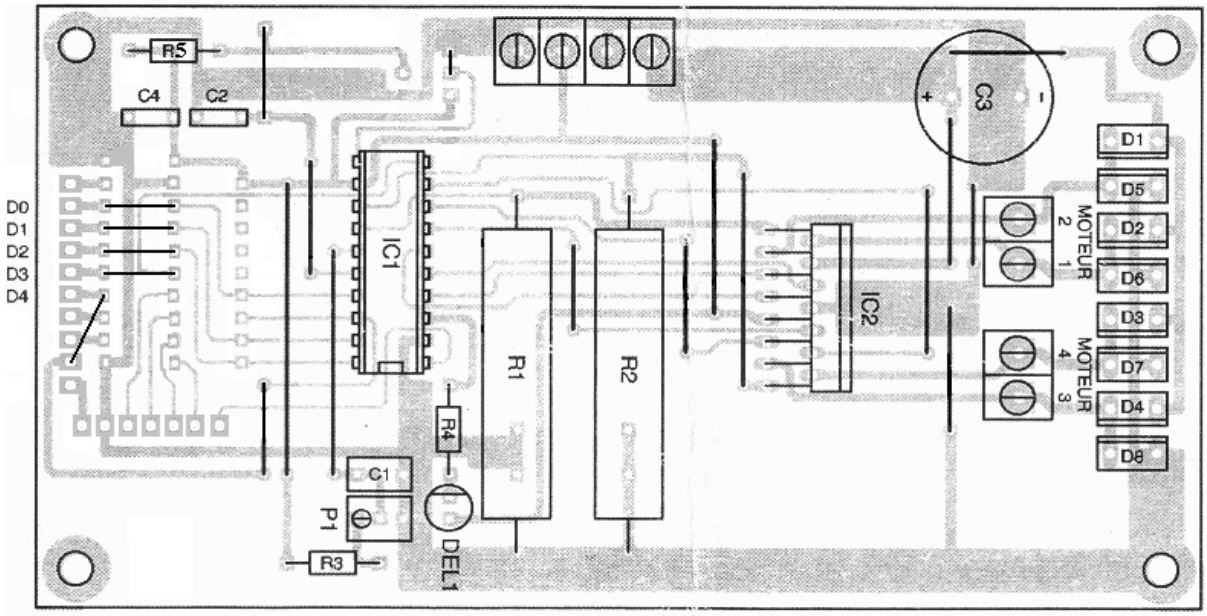


Fig. IV.6: Implémentation des composants de la carte de commande moteur.

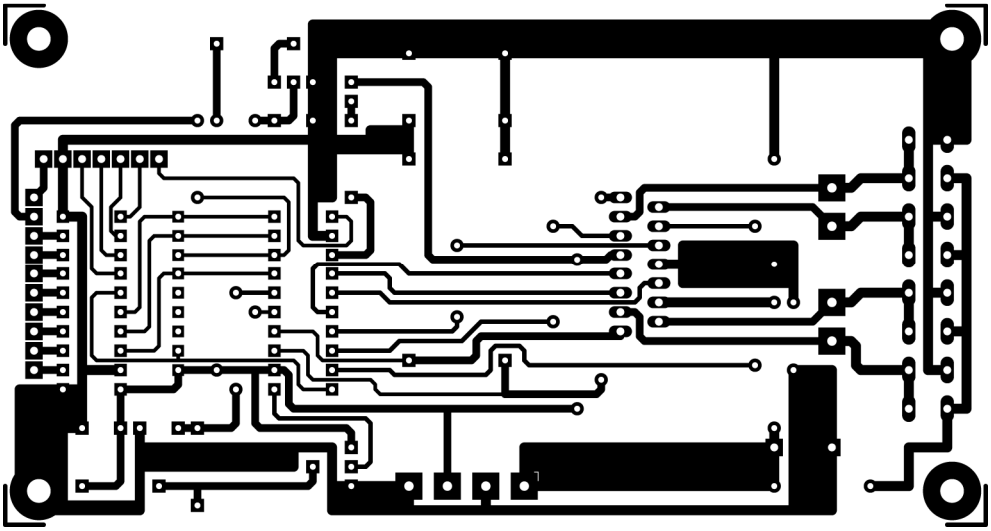


Fig. IV.7: Circuit imprimé de la carte de commande moteur.

IV.2. Tests et applications pratiques

Afin de finaliser notre travail on procède aux différents tests nécessaires qui confirment les résultats de calcul théoriques et ainsi nous permettre une opération de fraisage sur bois.

IV.2.1. Test sur l'exactitude du nombre de pas

Ce test consiste à vérifier le bon fonctionnement de la partie du programme qui s'occupe du calcul du nombre de pas et des différentes communications utilisées.

Pour cela, on a écrit une commande dont on connait le nombre de pas qu'elle doit effectuer, puis on a procédé à la vérification de son exactitude en sortie sur un moteur à vide pour éviter toute contrainte mécanique.

IV.2.2. Test sur l'exactitude de la fréquence de l'exécution des pas

Ce test consiste à vérifier le bon fonctionnement de la partie du programme qui s'occupe du calcul de la fréquence et des différentes communications utilisées.

Pour cela, on a procédé de la même méthode que la précédente sauf qu'à la sortie on a chronométré le temps de la manœuvre.

IV.2.3. Test sur la précision

Ce test consiste à connaitre la précision mécanique de notre fraiseuse à commande numérique.

Pour cela, on a écrit une commande dont la sortie doit effectuer un mouvement linéaire d'une valeur connue en millimètres, puis on à procédé à la vérification a l'aide du pied à coulisse en mesurent la distance parcourue. On a obtenu un résultat de l'ordre de 0.5mm.

IV.3. Mise en application

Le bon déroulement des tests nous permet de finaliser notre travail avec l'application d'une opération d'usinage.

Vu le type de fraises dont on dispose et l'absence du système d'arrosage sur notre machine, on est ramené à opter pour usiner du bois (tout type de bois).On a choisi de réaliser deux formes géométriques qui illustrent les différents mouvements possibles sur notre machine et qui sont le carré et le cercle.
Les différentes étapes sont décrites ci dessous :

✓ Dessin de la pièce à usiner

Le dessin se fait sur le logiciel mastercamv9 conçu pour le dessin et la génération du fichier de commandes G-Code.

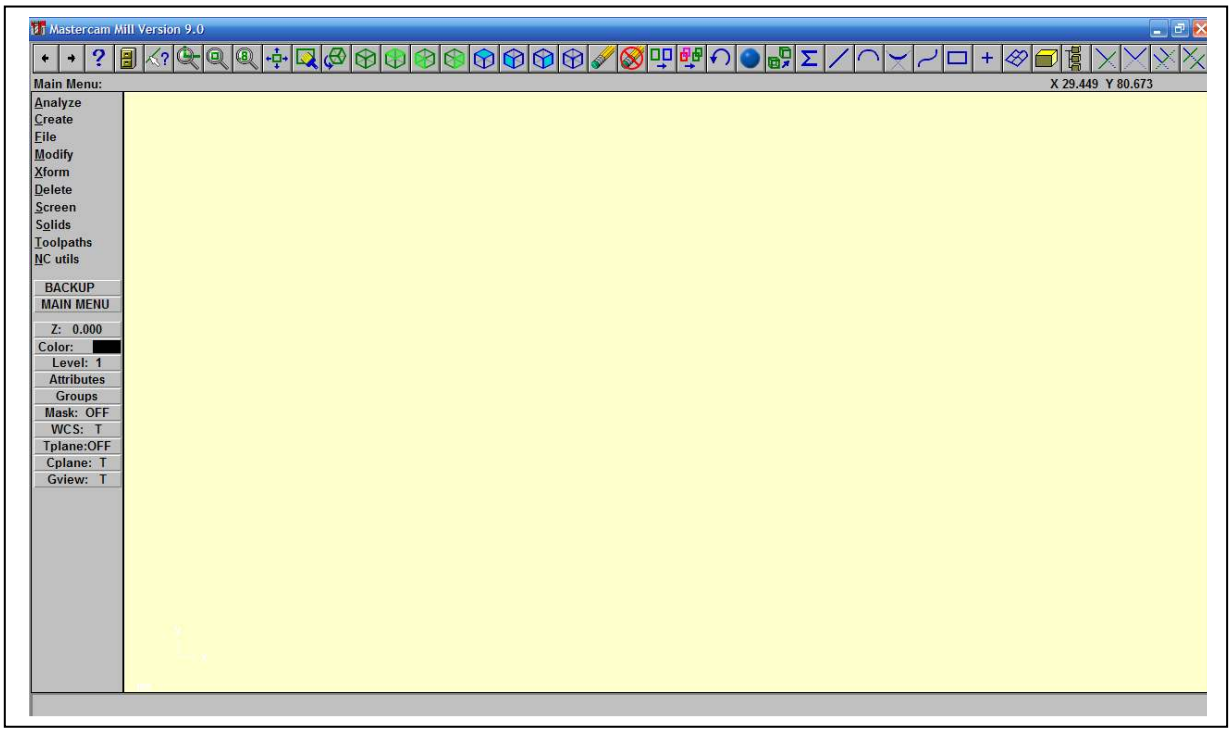


Fig. IV.8 : Vue de l'interface de travail du logiciel mastercamv9.

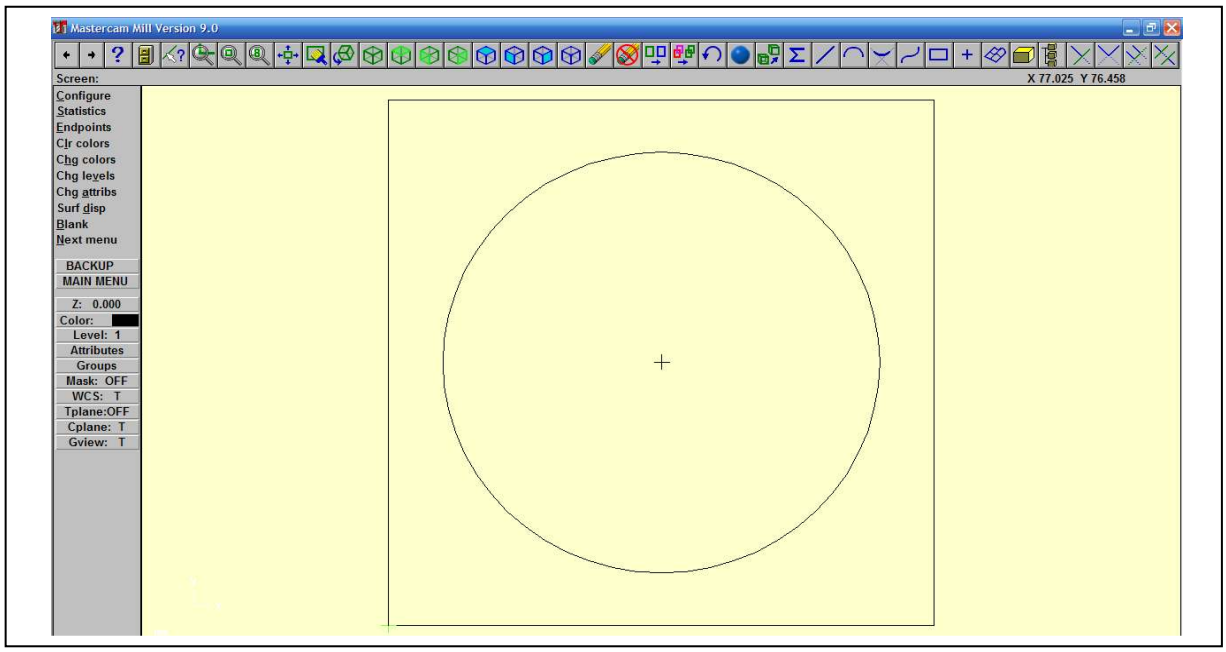


Fig. IV.9 : Vue du dessin de la pièce à usiner sur mastercamv9.

✓ Chargement du fichier dans l'interpréteur

Une fois le fichier G-code généré, on le charge dans notre interpréteur.

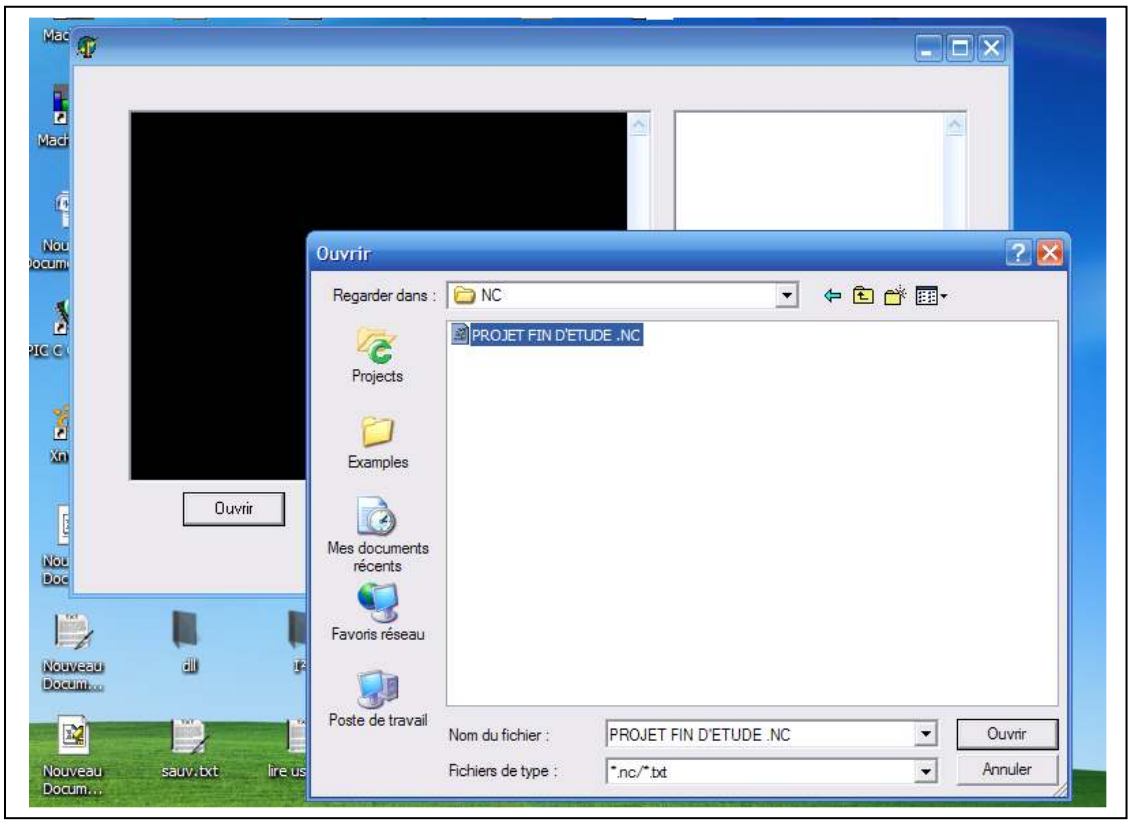


Fig. IV.10: Vue de la fenêtre du chargement du fichier G-Code dans l'interpréteur.

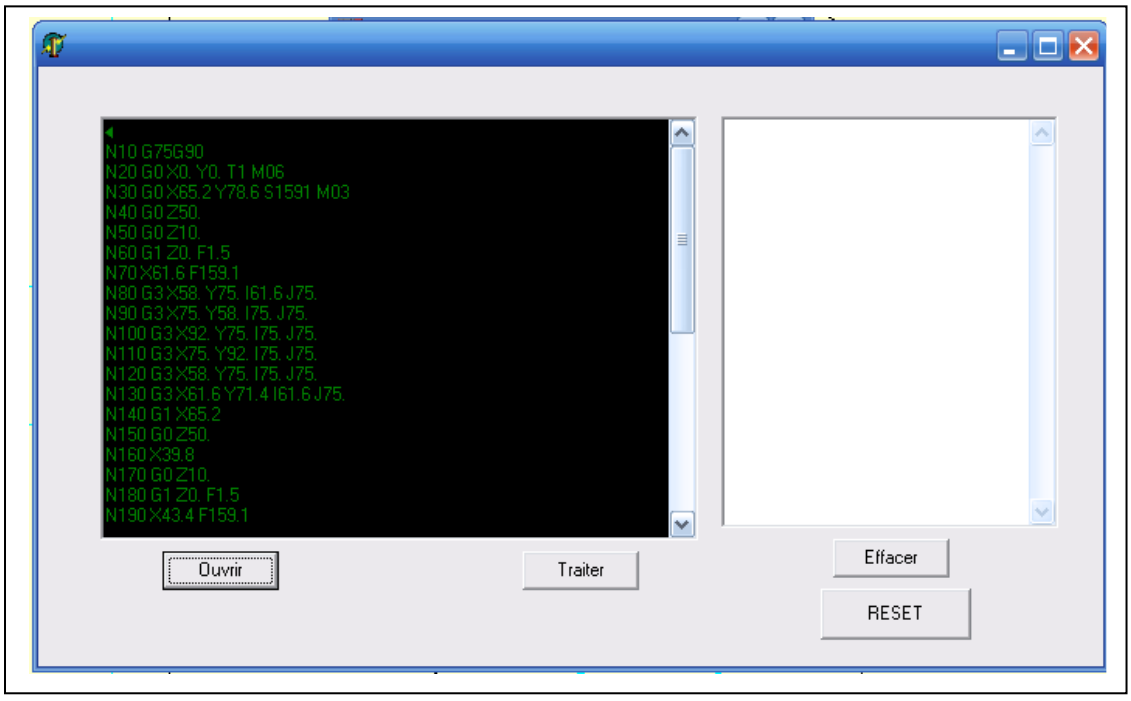


Fig. IV.11: Vue du fichier G-Code chargé dans l'interpréteur.

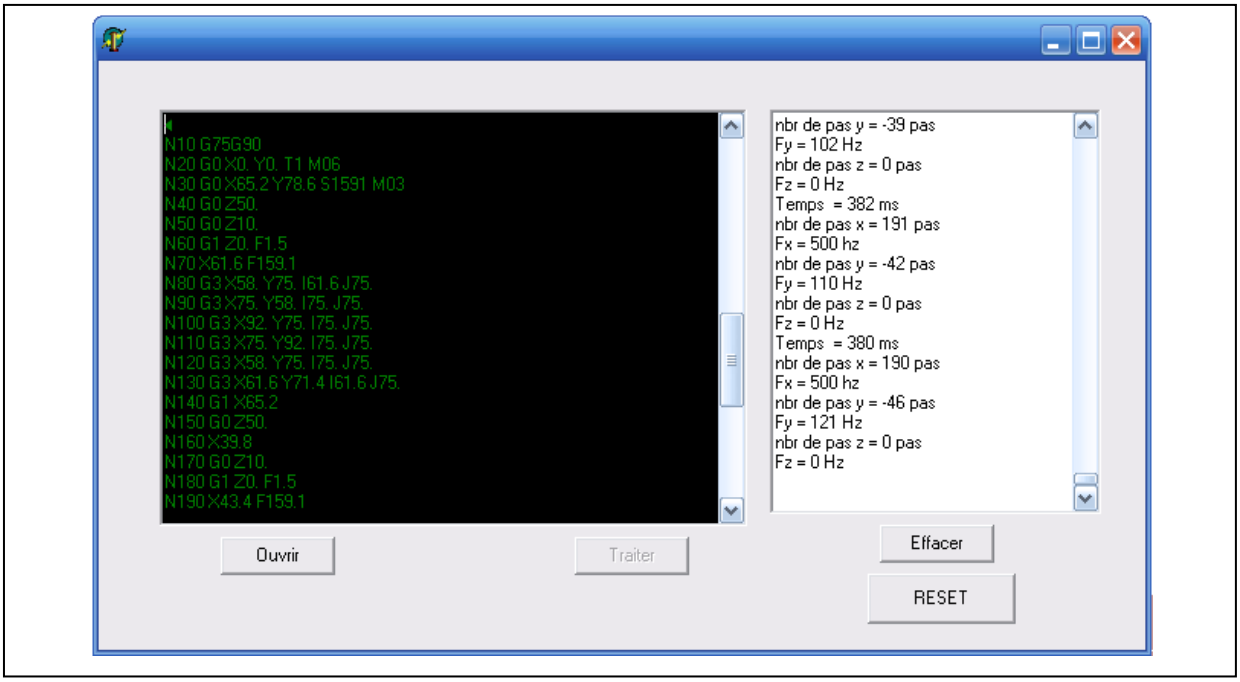


Fig. IV.12: Vue de l'interpréteur en cours de traitement.

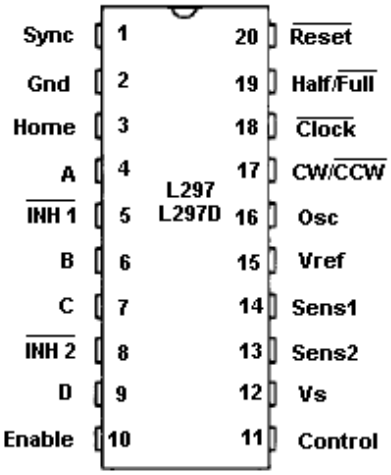
Conclusion Générale

Ce projet nous a permis de découvrir un domaine qui nous était méconnu, à savoir la mécatronique, ainsi que la mise en application des connaissances théoriques acquises durant notre formation. La conception de notre fraiseuse à commande numérique(CNC) nous a imposé le traitement de problèmes d'ordre mécanique, électronique et informatique.

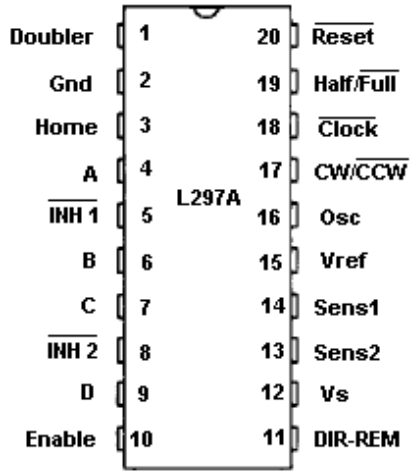
Parmi les perspectives envisagées à l'amélioration des performances de la machine, nous proposons d'intégrer plus de fonctionnalités telles que l'arrosage automatique, le changeur d'outils et les axes rotatifs.

Pour conclure, nous espérons que ce modeste travail pourra servir de référence aux travaux futures des prochaines promotions et les inciter à s'intéresser d'avantage au coté pratique de l'électronique.

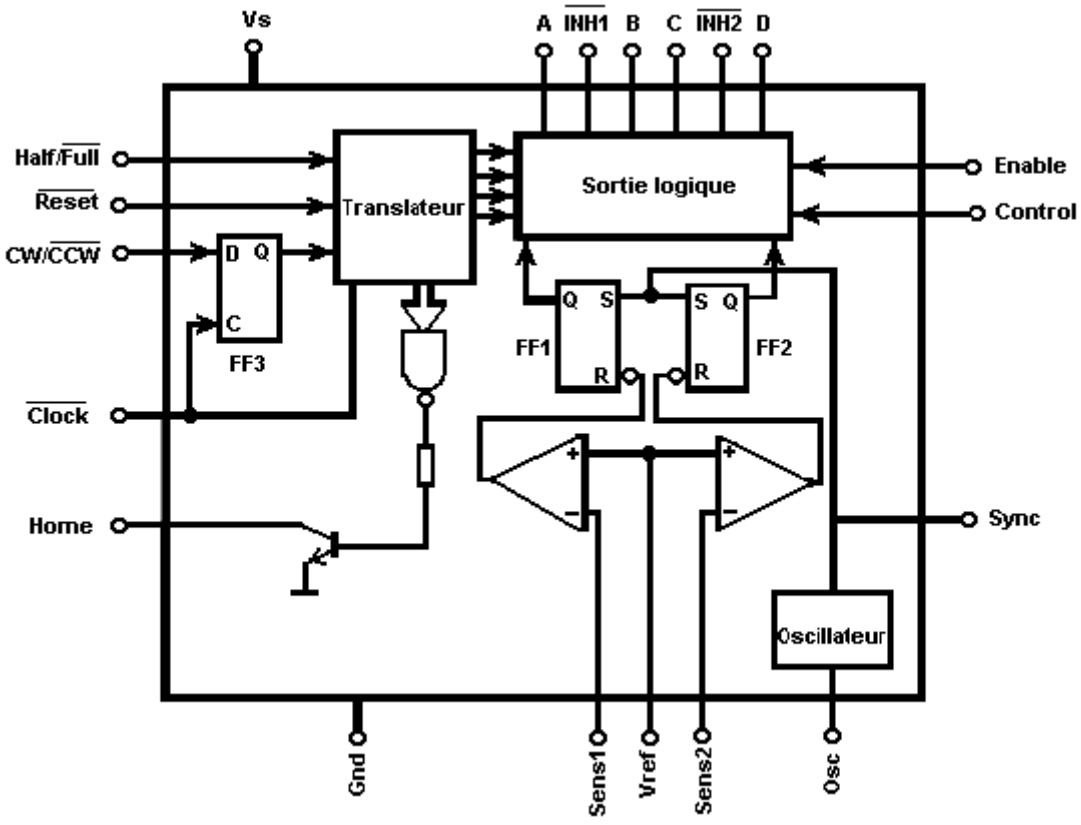
Brochage du circuit L297 et du L297D



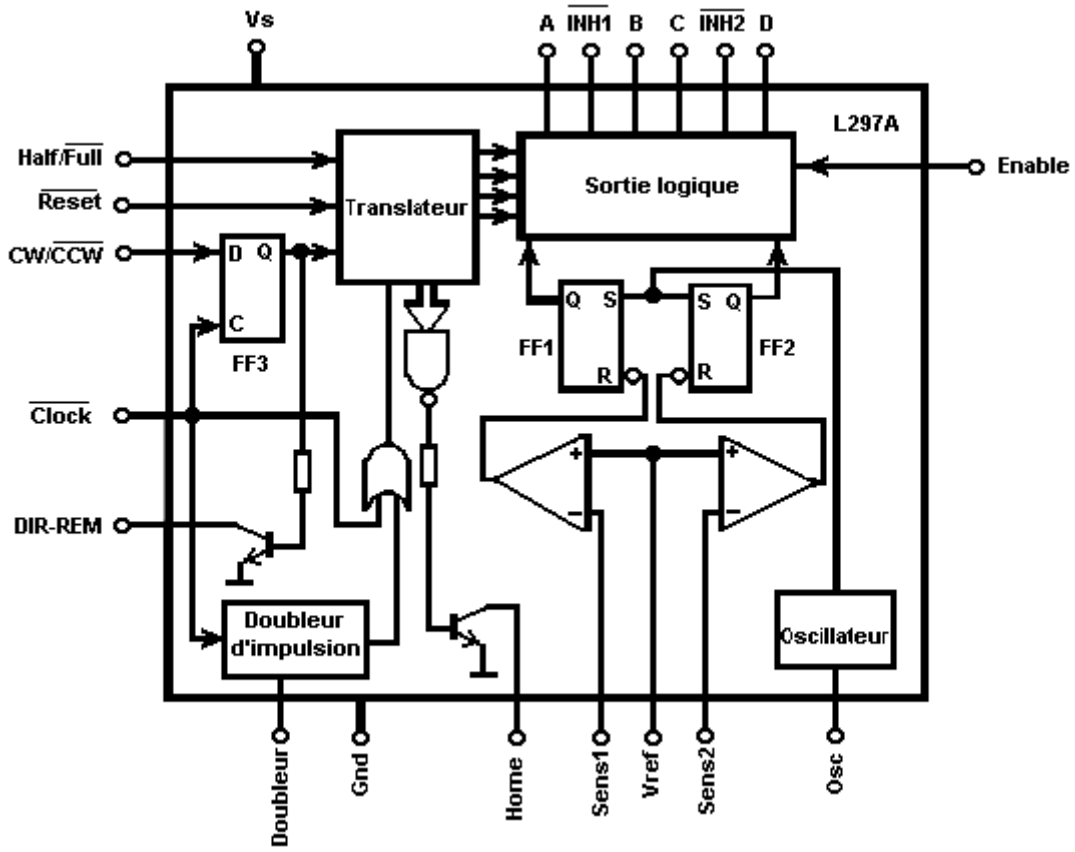
Brochage du circuit L297A



Structure interne du L297 et du L297D



Structure interne du L297A



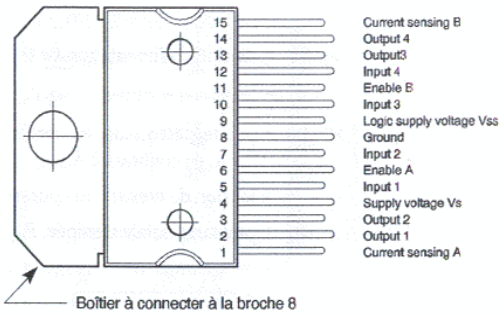
Fonctions des broches du L297

N°	NOM	FONCTION
1	*SYNC*	**C'est une sortie de l'oscillateur interne qui est reliée aux autres pins SYNC des autres L297 utilisés en parallèle.**
2	*GND*	**La masse du circuit.**
3	*HOME*	**Sortie indiquant l'état initial ABCD = 0101. Le transistor interne du circuit est à collecteur ouvert pour cette position.**
4	*A*	**Sortie de la phase A du moteur utilisée par l'étage de puissance.**
5	$\overline{\text{INH1}}$	**Un niveau bas implique la désactivation des sorties A et B. Pour la commande bipolaire, cette entrée permet la décharge rapide du courant lorsque les bobinages sont désalimentés.**
6	*B*	**Sortie de la phase B du moteur, utilisée par l'étage de puissance.**
7	*C*	**Sortie de la phase C du moteur, utilisée par l'étage de puissance.**
8	$\overline{INH2}$	**Un niveau bas implique la désactivation des sorties C et D. Pour la commande bipolaire, cette entrée permet la décharge rapide du courant lorsque les bobinages sont désalimentés.**
9	*D*	**Sortie de la phase D du moteur, utilisée par l'étage de puissance.**
10	*ENABLE*	**Entrée de validation. Les sorties A, B, C, $\overline{\text{INH1}}$, $\overline{INH2}$ sont mis à l'état bas lorsque cette entrée est à zéro.**
11	*CONTRÔLE*	**Quand un niveau haut est appliqué à cette entrée de contrôle ; le hacheur agit sur les sorties A, B, C et D. Sinon, il agit sur les $\overline{\text{INH1}}$ $\overline{INH2}$**

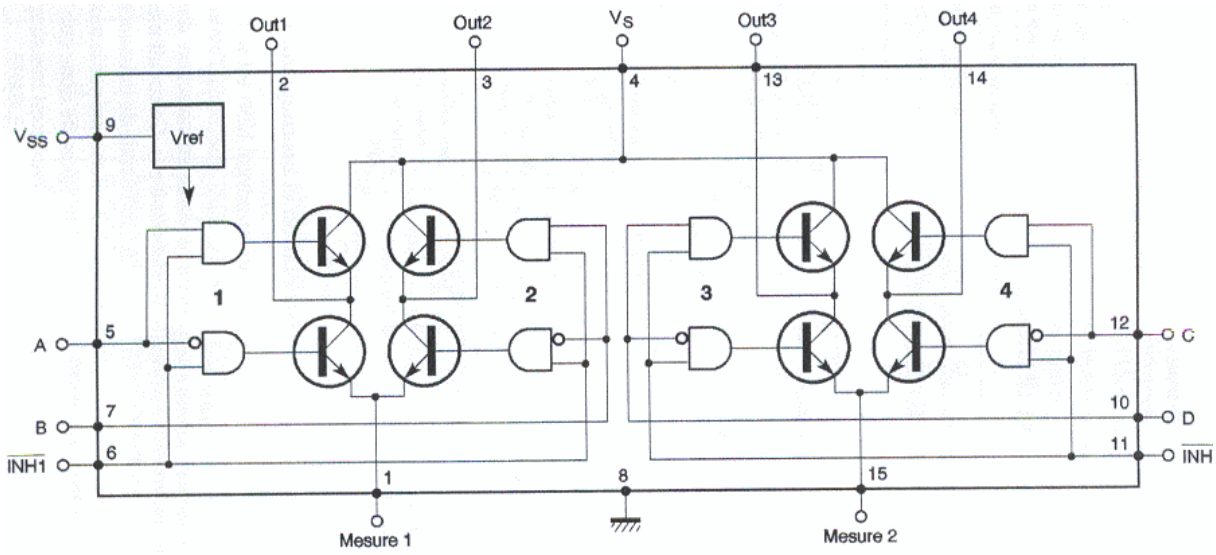
		sorties INH1 et INH2.
12	*Vs*	**L'alimentation +5V.**
13	*SENS2*	**Entrée de contrôle du courant traversant les bobinages C et D.**
14	*SENS1*	**Entrée de contrôle du courant traversant les bobinages A et B.**
15	*Vref*	**Entrée sur laquelle est appliquée la tension de référence qui détermine la valeur maximale du courant.**
16	*OSC*	**Entrée reliée à un oscillateur *RC* qui détermine la fréquence de fonctionnement du hacheur.**
17	*$\overline{CW/CCW}$*	**Entrée de contrôle du sens de rotation de l'arbre du moteur.**
18	*$\overline{CLOCK}$*	**Entrée d'horloge qui permet l'avance des pas sur front descendant.**
19	*$\overline{HALF/FULL}$*	**Entrée de sélection du mode de fonctionnement. Soit en pas, soit en demi pas.**
20	*$\overline{RESET}$*	**Entrée de remise à zéro. Après un Reset on retrouve l'état HOME(0101).**

Fonction et brochage du circuit intégré L298

N° de la broche	Nom	Fonction
1 et 15	SENSE A et SENSE B	Entre les deux broches et la masse doivent être connectées des résistances permettant de contrôler le courant circulant dans les bobinages du moteur.
2 et 3	OUT1 et OUT2	Sorties du pont A. Le courant débité par cet étage est commandé par la broche 1.
4	Vs	Entrée de la tension d'alimentation des étages de puissance. Un condensateur de 100 nF doit être connecté entre cette broche et la masse.
5 et 7	INPUT I et INPUT 2	Entrées compatibles TTL du pont A.
6 et 11	ENABLE A et ENABLE B	Entrées compatibles TTL permettant de valider les ponts A et B. Un niveau bas appliqué sur ces broches met les étages de puissance au repos.
5	GND	Masse du circuit.
9	Vss	Entrée de la tension d'alimentation de la partie logique du circuit. Un condensateur de 100 nF doit être connecté entre cette broche et la masse.
10 et 12	INPUT 3 et INPUT 4	Entrées compatibles TTL du pont B.
13 et 14	OUT3 et OUT4	Sorties du pont B. Le courant débité par cet étage est commandé par la broche 15.



Constitution interne du circuit intégré L298



Moteur pas à pas modèle 23LM-C « ASTROSYN » :

- BIPOLAIRE – 2 PHASES quatre fils
- Courant par phase : 2 A @ 2,6 V
- Couple : 61,4 Ncm
- Dimensions : Ø 56,4 x 56,4 x 49,5 (hors axe)
- Axe : Ø 6,35 x 30 mm
- Entraxe de fixation : 47,14 x 47,14 mm
- 200 pas / tour

Documents téchniques relatifs au PIC 18F2550

MICROCHIP

PIC18F2455/2550/4455/4550

28/40/44-Pin, High-Performance, Enhanced Flash, USB Microcontrollers with nanoWatt Technology

Universal Serial Bus Features:

- USB V2.0 Compliant
- Low Speed (1.5 Mb/s) and Full Speed (12 Mb/s)
- Supports Control, Interrupt, Isochronous and Bulk Transfers
- Supports up to 32 Endpoints (16 bidirectional)
- 1-Kbyte Dual Access RAM for USB
- On-Chip USB Transceiver with On-Chip Voltage Regulator
- Interface for Off-Chip USB Transceiver
- Streaming Parallel Port (SPP) for USB streaming transfers (40/44-pin devices only)

Power-Managed Modes:

- Run: CPU on, peripherals on
- Idle: CPU off, peripherals on
- Sleep: CPU off, peripherals off
- Idle mode currents down to 5.8 µA typical
- Sleep mode currents down to 0.1 µA typical
- Timer1 Oscillator: 1.1 µA typical, 32 kHz, 2V
- Watchdog Timer: 2.1 µA typical
- Two-Speed Oscillator Start-up

Flexible Oscillator Structure:

- Four Crystal modes, including High Precision PLL for USB
- Two External Clock modes, up to 48 MHz
- Internal Oscillator Block:
  - 8 user-selectable frequencies, from 31 kHz to 8 MHz
  - User-tunable to compensate for frequency drift
- Secondary Oscillator using Timer1 @ 32 kHz
- Dual Oscillator options allow microcontroller and USB module to run at different clock speeds
- Fail-Safe Clock Monitor:
  - Allows for safe shutdown if any clock stops

Peripheral Highlights:

- High-Current Sink/Source: 25 mA/25 mA
- Three External Interrupts
- Four Timer modules (Timer0 to Timer3)
- Up to 2 Capture/Compare/PWM (CCP) modules:
  - Capture is 16-bit, max. resolution 5.2 ns (TCY/16)
  - Compare is 16-bit, max. resolution 83.3 ns (TCY)
  - PWM output: PWM resolution is 1 to 10-bit
- Enhanced Capture/Compare/PWM (ECCP) module:
  - Multiple output modes
  - Selectable polarity
  - Programmable dead time
  - Auto-shutdown and auto-restart
- Enhanced USART module:
  - LIN bus support
- Master Synchronous Serial Port (MSSP) module supporting 3-wire SPI (all 4 modes) and I^2C™ Master and Slave modes
- 10-bit, up to 13-channel Analog-to-Digital Converter module (A/D) with Programmable Acquisition Time
- Dual Analog Comparators with Input Multiplexing

Special Microcontroller Features:

- C Compiler Optimized Architecture with optional Extended Instruction Set
- 100,000 Erase/Write Cycle Enhanced Flash Program Memory typical
- 1,000,000 Erase/Write Cycle Data EEPROM Memory typical
- Flash/Data EEPROM Retention: > 40 years
- Self-Programmable under Software Control
- Priority Levels for Interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
  - Programmable period from 41 ms to 131s
- Programmable Code Protection
- Single-Supply 5V In-Circuit Serial Programming™ (ICSP™) via two pins
- In-Circuit Debug (ICD) via two pins
- Optional dedicated ICD/ICSP port (44-pin devices only)
- Wide Operating Voltage Range (2.0V to 5.5V)

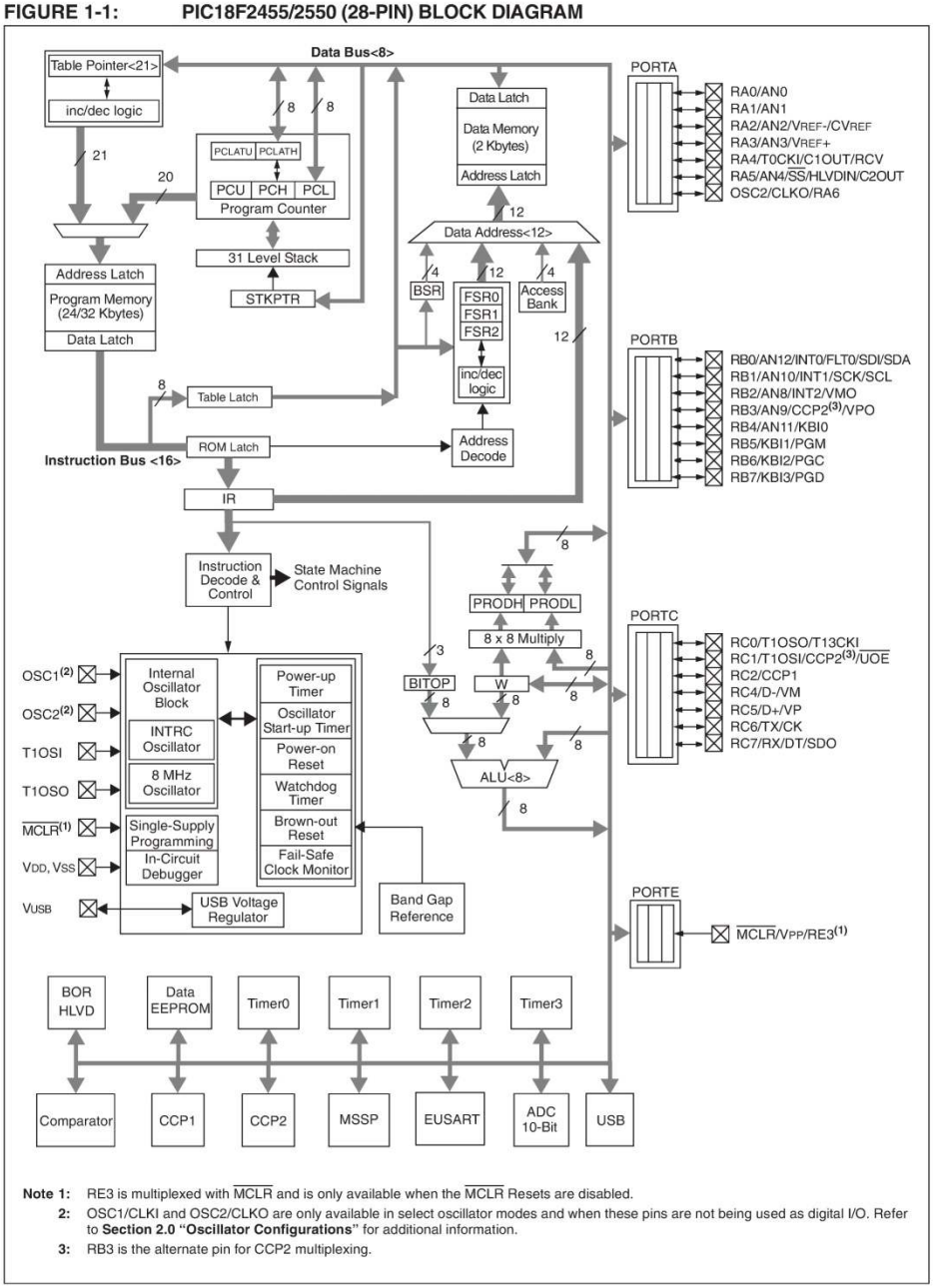
Device	Program Memory		Data Memory		I/O	10-Bit A/D (ch)	CCP/ECCP (PWM)	SPP	MSSP		EAUSART	Comparators	Timers 8/16-Bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)					SPI	Master I^2C™			
PIC18F2455	24K	12288	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F2550	32K	16384	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F4455	24K	12288	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3
PIC18F4550	32K	16384	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3

TABLE 1-1: DEVICE FEATURES

Features	PIC18F2455	PIC18F2550	PIC18F4455	PIC18F4550
Operating Frequency	DC – 48 MHz	DC – 48 MHz	DC – 48 MHz	DC – 48 MHz
Program Memory (Bytes)	24576	32768	24576	32768
Program Memory (Instructions)	12288	16384	12288	16384
Data Memory (Bytes)	2048	2048	2048	2048
Data EEPROM Memory (Bytes)	256	256	256	256
Interrupt Sources	19	19	20	20
I/O Ports	Ports A, B, C, (E)	Ports A, B, C, (E)	Ports A, B, C, D, E	Ports A, B, C, D, E
Timers	4	4	4	4
Capture/Compare/PWM Modules	2	2	1	1
Enhanced Capture/ Compare/PWM Modules	0	0	1	1
Serial Communications	MSSP, Enhanced USART	MSSP, Enhanced USART	MSSP, Enhanced USART	MSSP, Enhanced USART
Universal Serial Bus (USB) Module	1	1	1	1
Streaming Parallel Port (SPP)	No	No	Yes	Yes
10-Bit Analog-to-Digital Module	10 Input Channels	10 Input Channels	13 Input Channels	13 Input Channels
Comparators	2	2	2	2
Resets (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT
Programmable Low-Voltage Detect	Yes	Yes	Yes	Yes
Programmable Brown-out Reset	Yes	Yes	Yes	Yes
Instruction Set	75 Instructions; 83 with Extended Instruction Set enabled	75 Instructions; 83 with Extended Instruction Set enabled	75 Instructions; 83 with Extended Instruction Set enabled	75 Instructions; 83 with Extended Instruction Set enabled
Packages	28-pin PDIP 28-pin SOIC	28-pin PDIP 28-pin SOIC	40-pin PDIP 44-pin QFN 44-pin TQFP	40-pin PDIP 44-pin QFN 44-pin TQFP

PIC18F2455/2550/4455/4550

FIGURE 1-1: PIC18F2455/2550 (28-PIN) BLOCK DIAGRAM



Note 1: RE3 is multiplexed with $\overline{MCLR}$ and is only available when the $\overline{MCLR}$ Resets are disabled.

2: OSC1/CLKI and OSC2/CLKO are only available in select oscillator modes and when these pins are not being used as digital I/O. Refer to **Section 2.0 "Oscillator Configurations"** for additional information.

3: RB3 is the alternate pin for CCP2 multiplexing.

PIC18F2455/2550/4455/4550

5.3.5 SPECIAL FUNCTION REGISTERS

The Special Function Registers (SFRs) are registers used by the CPU and peripheral modules for controlling the desired operation of the device. These registers are implemented as static RAM in the data memory space. SFRs start at the top of data memory and extend downward to occupy the top segment of Bank 15, from F60h to FFFh. A list of these registers is given in Table 5-1 and Table 5-2.

The SFRs can be classified into two sets: those associated with the "core" device functionality (ALU, Resets and interrupts) and those related to the peripheral functions. The Reset and interrupt registers are described in their respective chapters, while the ALU's STATUS register is described later in this section. Registers related to the operation of a peripheral feature are described in the chapter for that peripheral.

The SFRs are typically distributed among the peripherals whose functions they control. Unused SFR locations are unimplemented and read as '0's.

TABLE 5-1: SPECIAL FUNCTION REGISTER MAP FOR PIC18F2455/2550/4455/4550 DEVICES

Address	Name	Address	Name	Address	Name	Address	Name	Address	Name
FFFh	TOSU	FDFh	INDF2(1)	FBFh	CCPR1H	F9Fh	IPR1	F7Fh	UEP15
FFEh	TOSH	FDEh	POSTINC2(1)	FBEh	CCPR1L	F9Eh	PIR1	F7Eh	UEP14
FFDh	TOSL	FDDh	POSTDEC2(1)	FBDh	CCP1CON	F9Dh	PIE1	F7Dh	UEP13
FFCh	STKPTR	FDCh	PREINC2(1)	FBCh	CCPR2H	F9Ch	—(2)	F7Ch	UEP12
FFBh	PCLATU	FDBh	PLUSW2(1)	FBBh	CCPR2L	F9Bh	OSCTUNE	F7Bh	UEP11
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	—(2)	F7Ah	UEP10
FF9h	PCL	FD9h	FSR2L	FB9h	—(2)	F99h	—(2)	F79h	UEP9
FF8h	TBLPTRU	FD8h	STATUS	FB8h	BAUDCON	F98h	—(2)	F78h	UEP8
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	ECCP1DEL	F97h	—(2)	F77h	UEP7
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	ECCP1AS	F96h	TRISE(3)	F76h	UEP6
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD(3)	F75h	UEP5
FF4h	PRODH	FD4h	—(2)	FB4h	CMCON	F94h	TRISC	F74h	UEP4
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB	F73h	UEP3
FF2h	INTCON	FD2h	HLVDCON	FB2h	TMR3L	F92h	TRISA	F72h	UEP2
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	—(2)	F71h	UEP1
FF0h	INTCON3	FD0h	RCON	FB0h	SPBRGH	F90h	—(2)	F70h	UEP0
FEFh	INDF0(1)	FCFh	TMR1H	FAFh	SPBRG	F8Fh	—(2)	F6Fh	UCFG
FEEh	POSTINC0(1)	FCEh	TMR1L	FAEh	RCREG	F8Eh	—(2)	F6Eh	UADDR
FEDh	POSTDEC0(1)	FCDh	T1CON	FADh	TXREG	F8Dh	LATE(3)	F6Dh	UCON
FECh	PREINC0(1)	FCCh	TMR2	FACh	TXSTA	F8Ch	LATD(3)	F6Ch	USTAT
FEBh	PLUSW0(1)	FCBh	PR2	FABh	RCSTA	F8Bh	LATC	F6Bh	UEIE
FEAh	FSR0H	FCAh	T2CON	FAAh	—(2)	F8Ah	LATB	F6Ah	UEIR
FE9h	FSR0L	FC9h	SSPBUF	FA9h	EEADR	F89h	LATA	F69h	UIE
FE8h	WREG	FC8h	SSPADD	FA8h	EEDATA	F88h	—(2)	F68h	UIR
FE7h	INDF1(1)	FC7h	SSPSTAT	FA7h	EECON2(1)	F87h	—(2)	F67h	UFRMH
FE6h	POSTINC1(1)	FC6h	SSPCON1	FA6h	EECON1	F86h	—(2)	F66h	UFRML
FE5h	POSTDEC1(1)	FC5h	SSPCON2	FA5h	—(2)	F85h	—(2)	F65h	SPPCON(3)
FE4h	PREINC1(1)	FC4h	ADRESH	FA4h	—(2)	F84h	PORTE	F64h	SPPEPS(3)
FE3h	PLUSW1(1)	FC3h	ADRESL	FA3h	—(2)	F83h	PORTD(3)	F63h	SPPCFG(3)
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC	F62h	SPPDATA(3)
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB	F61h	—(2)
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA	F60h	—(2)

Note 1: Not a physical register.
2: Unimplemented registers are read as '0'.
3: These registers are implemented only on 40/44-pin devices.

Documents téchniques relatifs au PIC 16F873A

MICROCHIP

PIC16F87XA

28/40/44-Pin Enhanced Flash Microcontrollers

Devices Included in this Data Sheet:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

High-Performance RISC CPU:

- Only 35 single-word instructions to learn
- All single-cycle instructions except for program branches, which are two-cycle
- Operating speed: DC – 20 MHz clock input
  DC – 200 ns instruction cycle
- Up to 8K x 14 words of Flash Program Memory, Up to 368 x 8 bytes of Data Memory (RAM), Up to 256 x 8 bytes of EEPROM Data Memory
- Pinout compatible to other 28-pin or 40/44-pin PIC16CXXX and PIC16FXXX microcontrollers

Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during Sleep via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- Synchronous Serial Port (SSP) with SPI™ (Master mode) and I^2C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) – 8 bits wide with external $\overline{RD}$, $\overline{WR}$ and $\overline{CS}$ controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

Analog Features:

- 10-bit, up to 8-channel Analog-to-Digital Converter (A/D)
- Brown-out Reset (BOR)
- Analog Comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (V_{REF}) module
  - Programmable input multiplexing from device inputs and internal voltage reference
  - Comparator outputs are externally accessible

Special Microcontroller Features:

- 100,000 erase/write cycle Enhanced Flash program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- Data EEPROM Retention > 40 years
- Self-reprogrammable under software control
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Single-supply 5V In-Circuit Serial Programming
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving Sleep mode
- Selectable oscillator options
- In-Circuit Debug (ICD) via two pins

CMOS Technology:

- Low-power, high-speed Flash/EEPROM technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Commercial and Industrial temperature ranges
- Low-power consumption

Device	Program Memory		Data SRAM (Bytes)	EEPROM (Bytes)	I/O	10-bit A/D (ch)	CCP (PWM)	MSSP		USART	Timers 8/16-bit	Comparators
	Bytes	# Single Word Instructions						SPI	Master I^2C			
PIC16F873A	7.2K	4096	192	128	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F874A	7.2K	4096	192	128	33	8	2	Yes	Yes	Yes	2/1	2
PIC16F876A	14.3K	8192	368	256	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F877A	14.3K	8192	368	256	33	8	2	Yes	Yes	Yes	2/1	2

1.0 DEVICE OVERVIEW

This document contains device specific information about the following devices:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

PIC16F873A/876A devices are available only in 28-pin packages, while PIC16F874A/877A devices are available in 40-pin and 44-pin packages. All devices in the PIC16F87XA family share common architecture with the following differences:

- The PIC16F873A and PIC16F874A have one-half of the total on-chip memory of the PIC16F876A and PIC16F877A
- The 28-pin devices have three I/O ports, while the 40/44-pin devices have five
- The 28-pin devices have fourteen interrupts, while the 40/44-pin devices have fifteen
- The 28-pin devices have five A/D input channels, while the 40/44-pin devices have eight
- The Parallel Slave Port is implemented only on the 40/44-pin devices

The available features are summarized in Table 1-1. Block diagrams of the PIC16F873A/876A and PIC16F874A/877A devices are provided in Figure 1-1 and Figure 1-2, respectively. The pinouts for these device families are listed in Table 1-2 and Table 1-3.

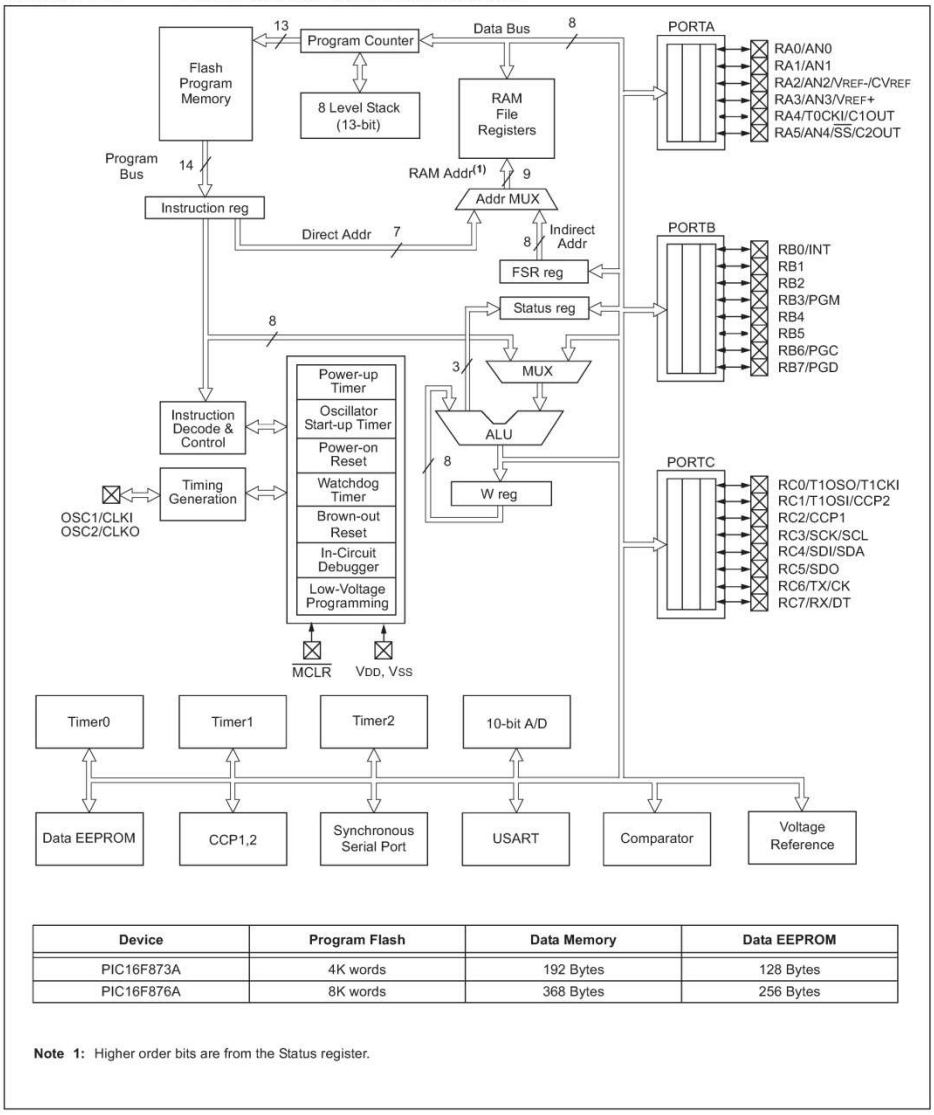
Additional information may be found in the PICmicro® Mid-Range Reference Manual (DS33023), which may be obtained from your local Microchip Sales Representative or downloaded from the Microchip web site. The Reference Manual should be considered a complementary document to this data sheet and is highly recommended reading for a better understanding of the device architecture and operation of the peripheral modules.

TABLE 1-1: PIC16F87XA DEVICE FEATURES

Key Features	PIC16F873A	PIC16F874A	PIC16F876A	PIC16F877A
Operating Frequency	DC – 20 MHz	DC – 20 MHz	DC – 20 MHz	DC – 20 MHz
Resets (and Delays)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)
Flash Program Memory (14-bit words)	4K	4K	8K	8K
Data Memory (bytes)	192	192	368	368
EEPROM Data Memory (bytes)	128	128	256	256
Interrupts	14	15	14	15
I/O Ports	Ports A, B, C	Ports A, B, C, D, E	Ports A, B, C	Ports A, B, C, D, E
Timers	3	3	3	3
Capture/Compare/PWM modules	2	2	2	2
Serial Communications	MSSP, USART	MSSP, USART	MSSP, USART	MSSP, USART
Parallel Communications	—	PSP	—	PSP
10-bit Analog-to-Digital Module	5 input channels	8 input channels	5 input channels	8 input channels
Analog Comparators	2	2	2	2
Instruction Set	35 Instructions	35 Instructions	35 Instructions	35 Instructions
Packages	28-pin PDIP 28-pin SOIC 28-pin SSOP 28-pin QFN	40-pin PDIP 44-pin PLCC 44-pin TQFP 44-pin QFN	28-pin PDIP 28-pin SOIC 28-pin SSOP 28-pin QFN	40-pin PDIP 44-pin PLCC 44-pin TQFP 44-pin QFN

PIC16F87XA

FIGURE 1-1: PIC16F873A/876A BLOCK DIAGRAM

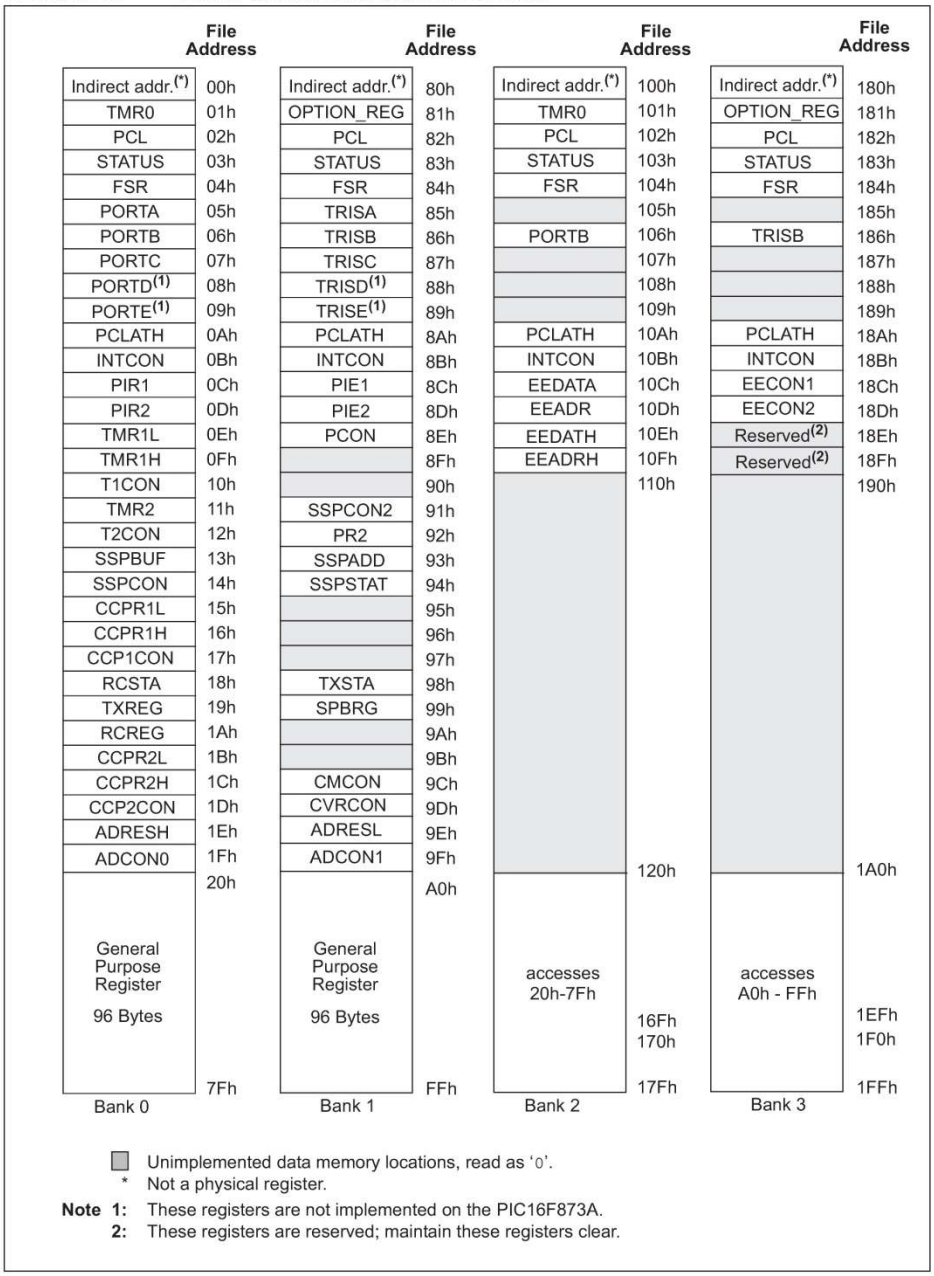


Device	Program Flash	Data Memory	Data EEPROM
PIC16F873A	4K words	192 Bytes	128 Bytes
PIC16F876A	8K words	368 Bytes	256 Bytes

Note 1: Higher order bits are from the Status register.

PIC16F87XA

FIGURE 2-4: PIC16F873A/874A REGISTER FILE MAP



Unimplemented data memory locations, read as '0'.
* Not a physical register.

Note 1: These registers are not implemented on the PIC16F873A.
2: These registers are reserved; maintain these registers clear.

- Patrice Oguic **«** *Moteurs pas à pas et PC » :* Editions Techniques et Scientifiques Françaises (ETSF), Paris France (2004)

- Alain Deboux *« S'initialiser à la programmation des PIC »* **:** Editions Techniques et Scientifiques Françaises (ETSF), Paris France (2001)

- Christian Tavernier « Programmation en C des PICs » : Edition DUNOD Collection Techn.et Ingénierie (2005).

- Dogan Ibrahim «Advanced PIC microcontroller projects in C: from USB to RTOS with the PIC18F series» newnes (2008)

INTERNET

- http://www.mastercam.com/
  Mastercam mill/design tutorial.

- http://www.ccsinfo.com/
  Manual CCS, INC.- C compiler Syntax and fonction.

- http://www.bigonoff.com
  Bigonoff « Première partie Pic16f84 ».
  Bigonoff « seconde partie la gamme mid-range par l'étude des16F87X ».
  Bigonoff «Cinquième partie la gamme high-end par l'étude des 18F ».

- http://www.microchip.com
  C18 Manual Microchip.

- http://www.linuxcnc.org (le langage RS274/NGC)

- http:// http://www.delphibasics.co.uk