Dans le chapitre précédent nous avons décrit toutes les étapes concernant la réalisation de la base mobile. Dans le chapitre présent nous présentons la partie logicielle réalisée dans notre travail.

Partie logiciel :

IV.1.La programmation du microcontrôleur PIC 16F877 :

IV.1.1.Le compilateur MIKRO C :

La programmation des microcontrôleurs PICs peut se faire par plusieurs langages informatiques tels que l'assembleur et le langage C qui sont les plus répondus dans les applications à base des PICs.

Dans notre projet nous avons opté pour le compilateur de MIKRO C qui est un compilateur en langage C (langage évolué) et qui permet d'intégrer certaines routines en assembleur. Ceci nous permet ainsi d'écrire les routines d'interruptions en assembleur afin de garantir un temps de traitement rapide qui nous garanti une commande en temps réel du robot.

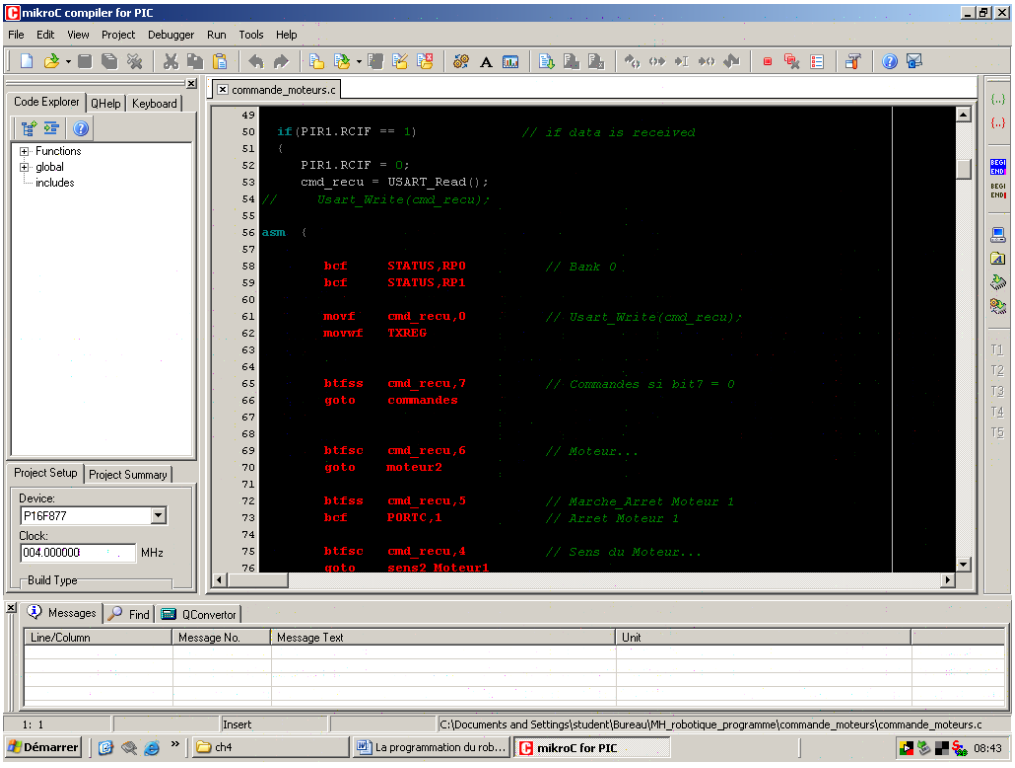

Figure.IV.1: mikroC

IV.1.2.Le logiciel de programmation du PIC –WINPIC :

WINPIC est un logiciel qui permet de transferer les programmes convertis en code hexadécimal vers la mémoire flash du microcontrôleur via le programmateur.

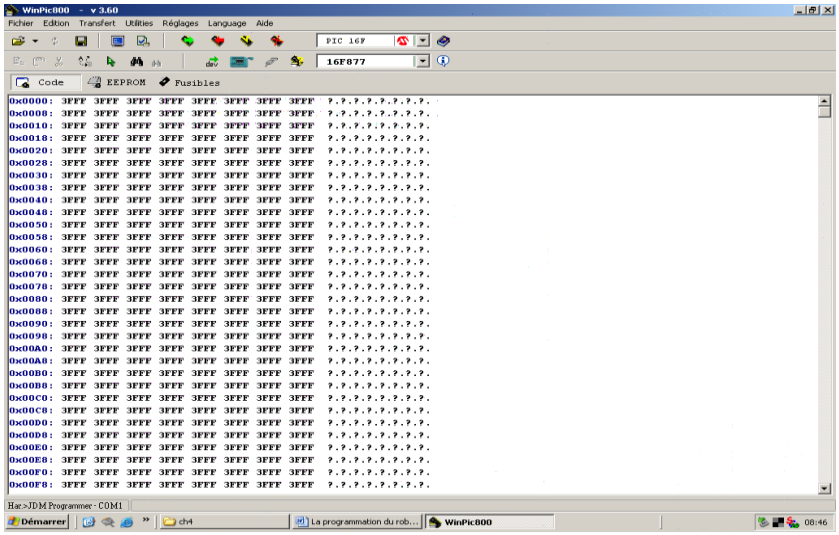

Figure.IV. 2: WinPic800

IV.1.3.L'éditeur de schéma PROTEUS :

Proteus est une suite de deux logiciels, Isis et Ares, Isis permet le dessin des schémas électroniques, la simulation en temps réel de façon dynamique et Ares permet de transformer les schémas en PCB (Tracé du circuit imprimé).

Proteus est un outil convivial facile à prendre en main. Il reconnaît et simule beaucoup de composants ainsi que de nombreux afficheurs, moteurs,…

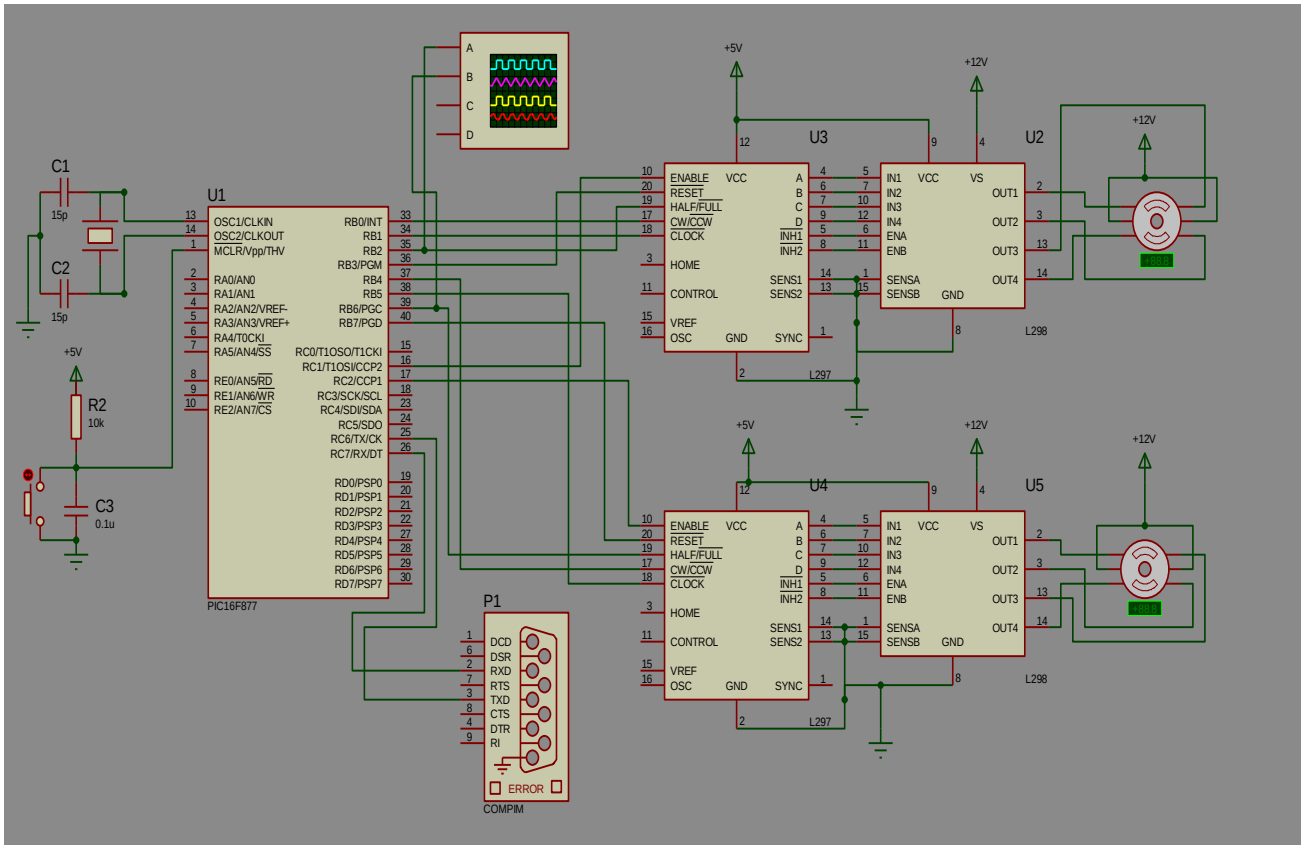

Figure. IV. 3: Le schéma de simulation de notre programme final

IV.1.4.Le programme du robot :

Le programme que nous avons implémenté dans notre robot se divise en deux parties :

- Un programme principal qui permet l'initialisation des entrées sorties du micro contrôleur et des variables utilisées dans le programme.
- Un sous programme de traitement des interruptions qui permet de gérer les interruptions de l'USART et du Timer1 utilisées dans notre programme.

IV.1.4.1.Le programme principale :

On peut présenter notre programme principal par l'organigramme suivant :

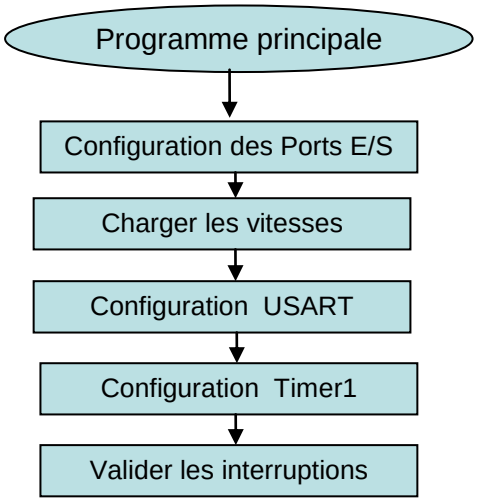

Figure.IV. 4: L'organigramme du programme principal

Pour minimiser le temps de transmisision et de traitement on a choisi d'utiliser 16 valeurs de vitesses possibles pour le robot. Ceci nous permet ainsi de réduire l'information de la vitesse transmise du PC vers le robot de 16 bits à seulement 4 bits qui indiquent l'indice de la vitesse choisie.

IV.1.4.2.Le sous programme de traitement des interruptions :

Ce programme va traiter deux interruptions :

- L'interruption provoquée par l'USART pour gérer la communication entre le robot et le PC. C'est l'interruption la plus prioritaire.
- L'interruption provoquée par TMR1 pour gérer la commande des deux moteurs.

On peut représenter le déroulement de ce programme par l'organigramme suivant :

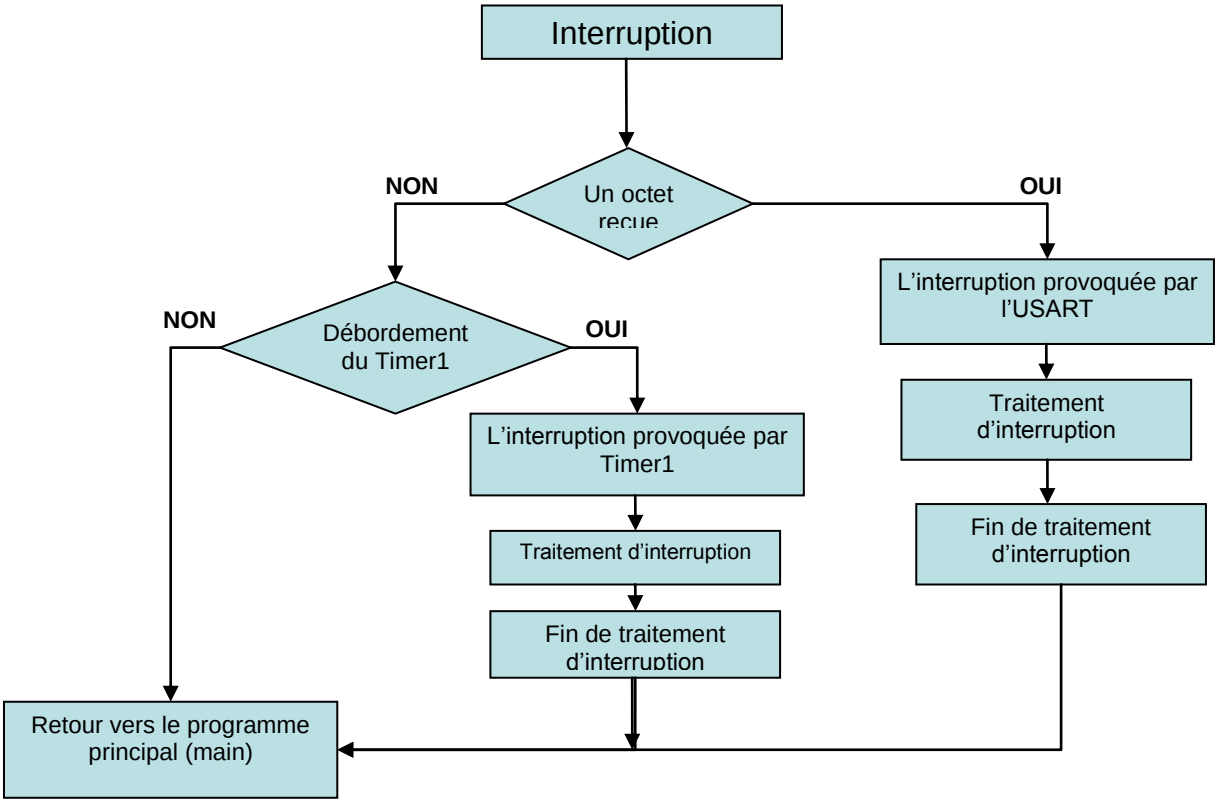

Figure.IV. 5: L'organigramme de sous programme de traitement des interruptions

IV.1.4.2.1.Traitement d'interruption provoquée par l'USART :

Afin d'optimiser la communication entre le PC et le robot on a essayé de réduire au maximum le protocole de communication utilisée en utilisant généralement 1 octet pour représenter une donnée.
La trame envoyée du pc au robot peut etre classée en 2 types de commande :
-Trame qui indique une vitesse
-Trame qui indique une commande que le robot doit éxécuter.

cmd_recu :

B7	B6	B5	B4	B3	B2	B1	B0

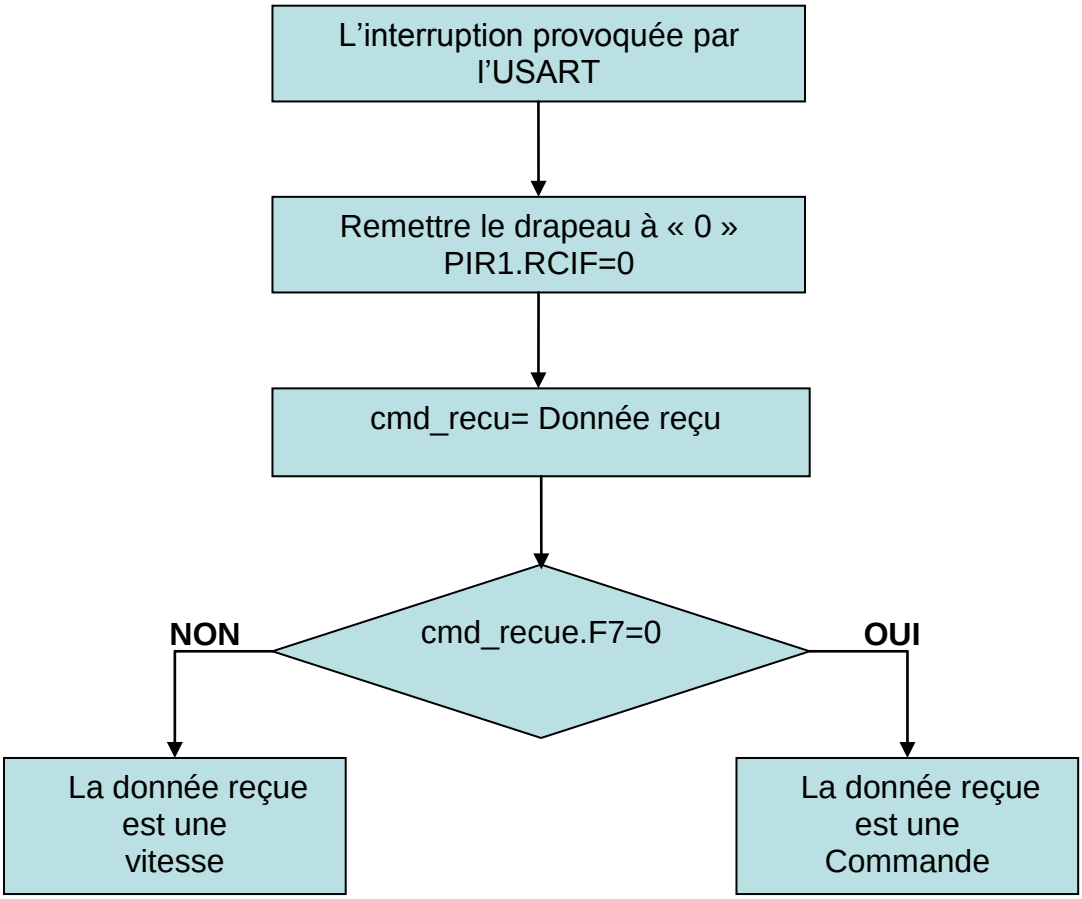

Figure.IV. 6: L'organigramme de sous programme de traitement d'interruption provoquer par l'USART

Le teste du 8éme bit de l'octet reçue permet le choix entre la vitesse et la commande c'est-à-dire :

Si B7=1 La donnée est une vitesse.

Sinon B7=0 La donnée est une commande.

IV.1.4.2.2.Le cas ou la donnée reçue est une commande :

Pour commander notre robot on a choisi deux modes de commandes :

-*Mode console de commande* : Consiste à commander directement le déplacement du robot via le PC.

-*Mode de commande par trajectoire* : Consiste à envoyer au robot les parametres d'une trajectoire rectiligne, le robot doit reproduire exactement la trajectoire.

Le choix du mode de la commande utilisée est indiqué dans le 7 éme bit (B6) de la commande envoyée par PC au robot.

L'organigramme suivant représente le programme de determination du mode de la commande choisie.

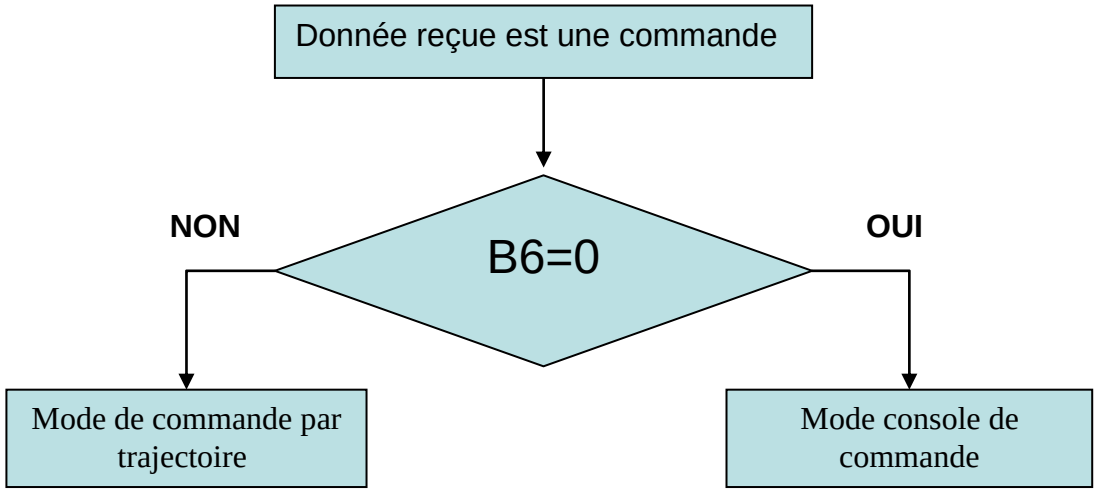

Figure.IV. 7: L'organigramme pour déterminer le mode de la commande choisie

On présente dans ce qui suit les programmes des deux modes de fonctionnement utilisés dans notre robot.

IV.1.4.2.2.1 Mode console de commande:

Mode console de commande consiste à envoyer d'abord la vitesse de chaque moteur au robot puis on envoi la commande de mise en marche.
Dans la commande de mise en marche on peut soit commander les deux moteurs en même temps soit commander uniquement un seul moteur.

Les différentes commandes de mise en marche sont determinées comme suit :

- cmd_recu = 00000001 → Arrêt totale (Arrêter les deux moteurs).
- cmd_recu = 00000010 → Marche (Démarrer les deux moteurs).
- cmd_recu = 00000100 → Marche 1 (Démarrer le moteur1 et arrêter le moteur2).
- cmd_recu = 00001000 → Marche 2 (Démarrer le moteur2 et arrêter le moteur1).

Pour la détermination du mode de fonctionnement nous avons appliqué l'organigramme suivant :

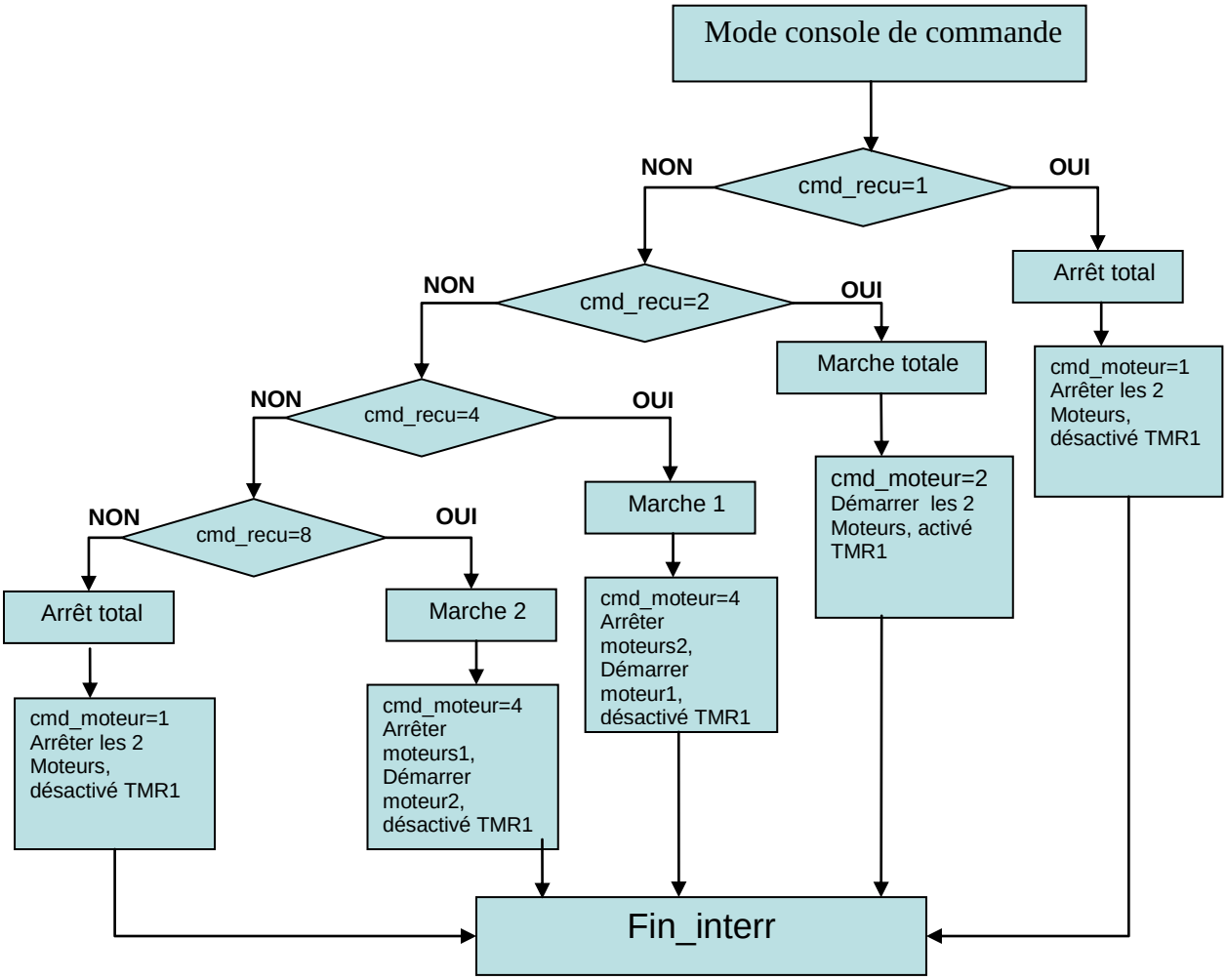

Figure.IV. 8: L'organigramme du mode console de commande

IV.1.4.2.2.2.Détermination de la vitesse :

Dans ce cas on va déterminer le sens de rotation et la mise en marche de chaque moteur et aussi la vitesse :

cmd_recu	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**

- B6 : Choix du moteur.
- B5 : Marche/Arrêt.
- B4 : Sens de rotation du moteur.
- B3-B2-B1-B0 : Vitesse.

Pour notre programme nous avons appliqué l'organigramme suivant :

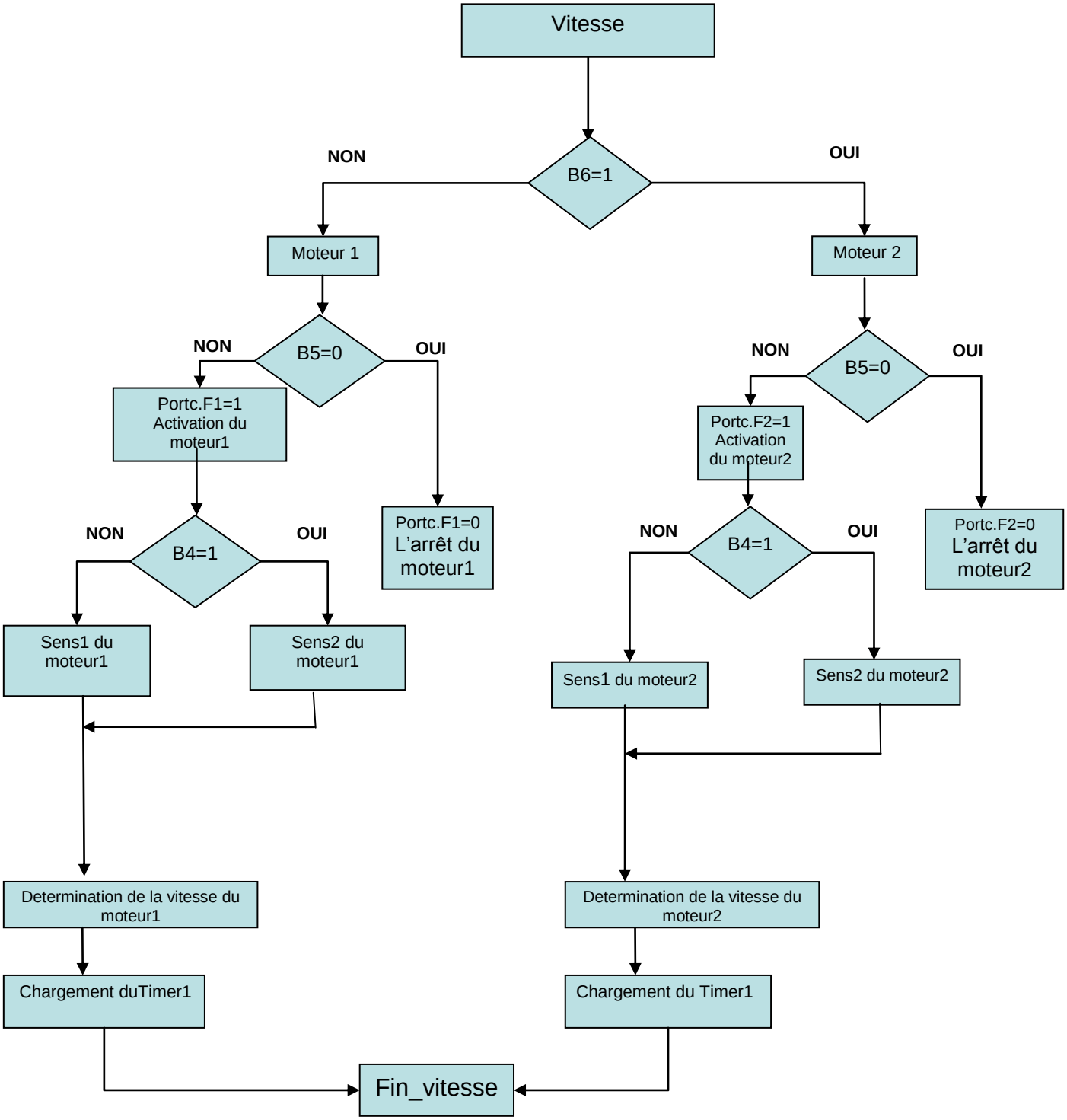

Figure.IV. 9: L'organigramme pour la détermination de la vitesse

IV.1.4.2.2.3.Gestion de l'interruption déclenchée par le Timer1 :

Le timer1 est attribué aux 2 moteurs. A chaque interruption on détermine le moteur qui sera attribué au timer1.

L'organigramme suivant présente l'algorithme utilisé pour la détermination la valeur qui sera chargée dans le timer1.

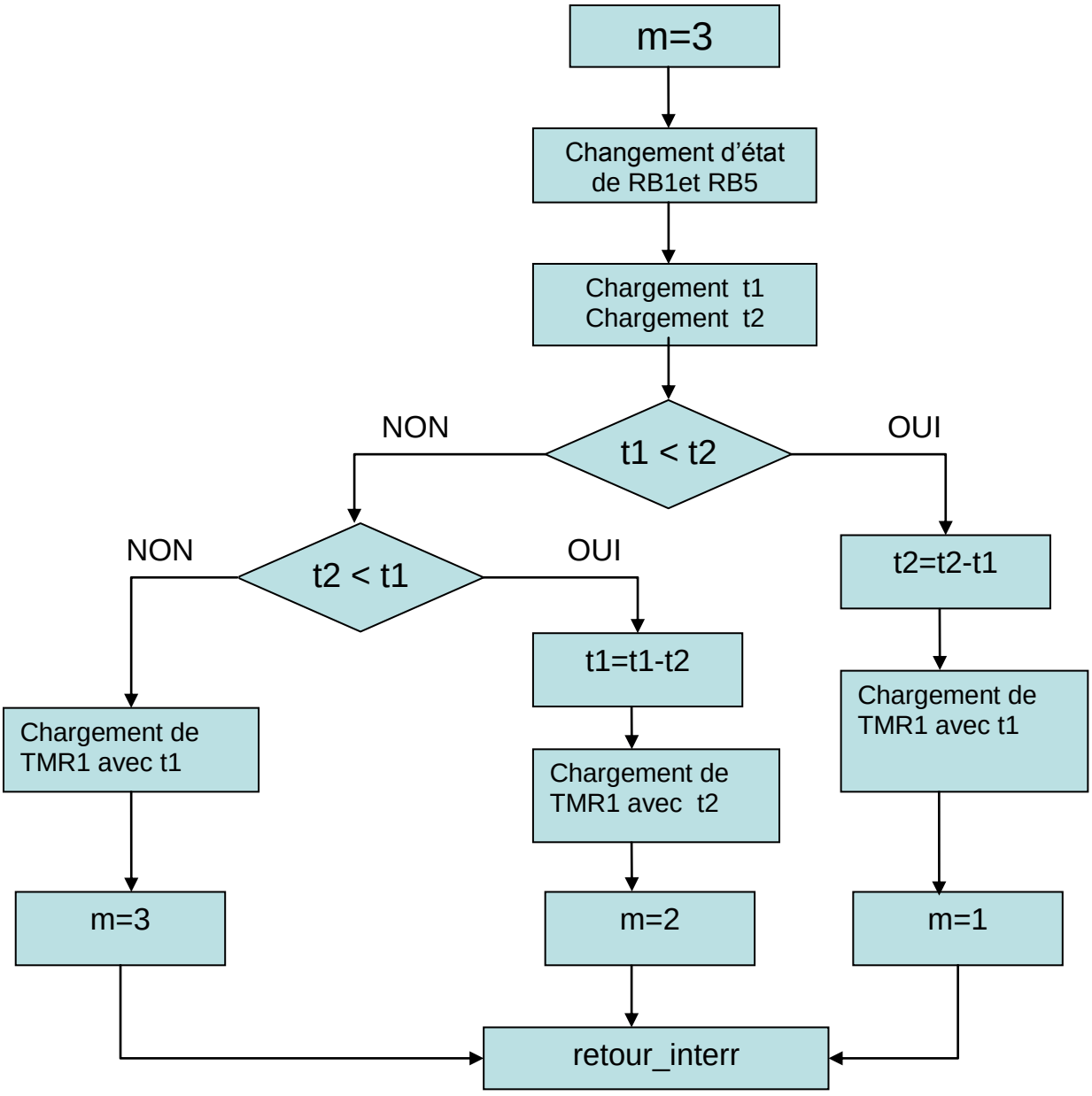

Figure.IV. 10: L'organigramme de gestion de l'interruption déclenchée par le Timer1 dans le mode console de commande

IV.1.4.2.1.2.2. Mode de commande par trajectoire :

Mode de commande par trajectoire consiste à envoyer la meme vitesse au deux moteurs ainsi que le sens de rotation de chacun dans une trame de 1 octet suivi par le nombre de pas (2octets) que le robot doit effectué et par une commande de mise en marche (1 octet) .
L'organigramme suivant présente l'algorithme utilisé pour le chargement de la vitesse le compteur

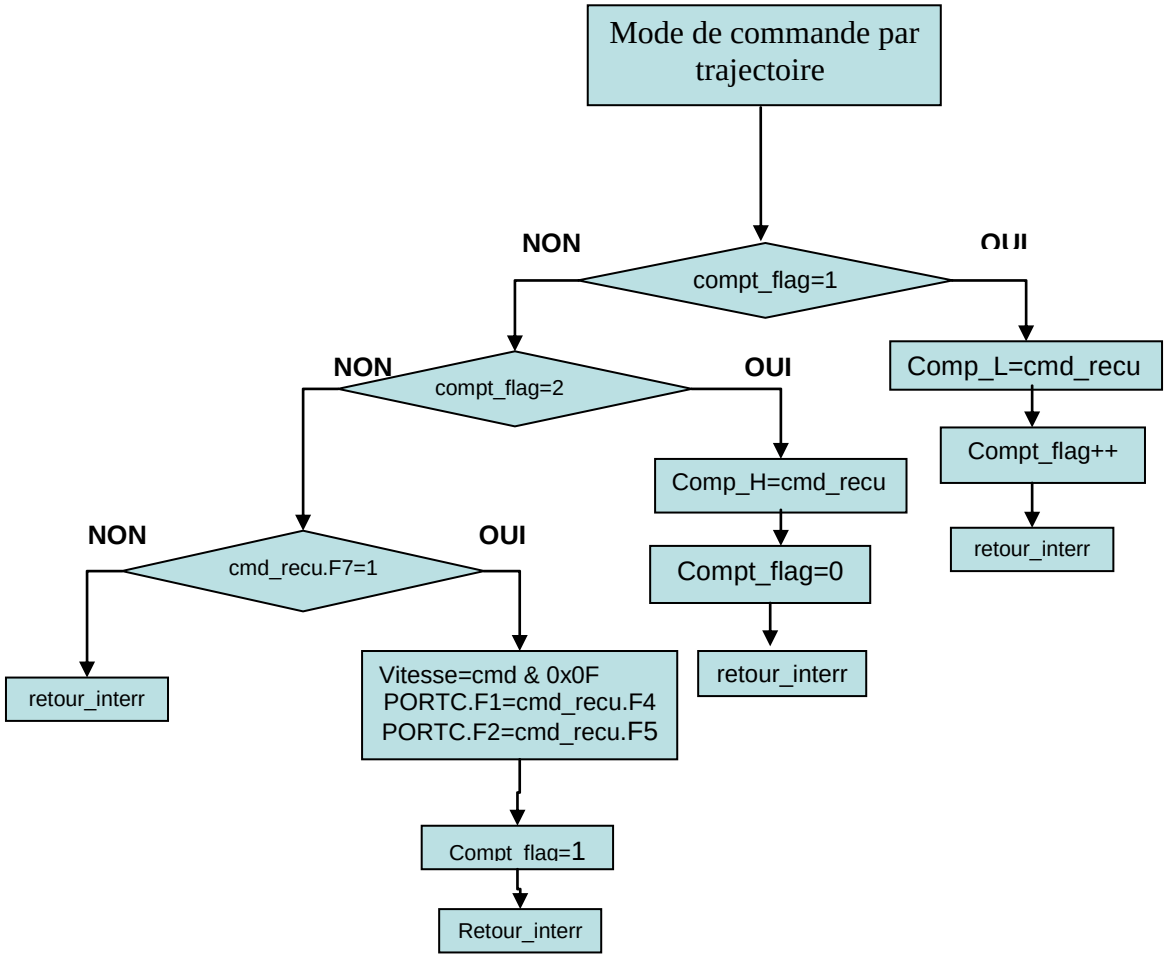

Figure.IV. 11: L'organigramme du mode de commande par trajectoire

A la réception de la vitesse transmise par le PC, la variable compt_flag ayant été initialisée par 0 sera mise à 1.Aprés la réception du poids faible du compteur la variable compt_flag sera mise à 2.
Après la réception du poids fort du compteur la variable compt_flag sera remise à 0.

IV.1.4.2.1.2.Gestion de l'interruption déclenchée par le Timer1 :

Un compteur de 16bits est utilisé pour exécuter le nombre de pas transmis par le PC. A chaque interruption le compteur est décrémenté. Lorsque le compteur atteint la valeur '0' le robot s'arrête et envoie le caractère'0' au PC.

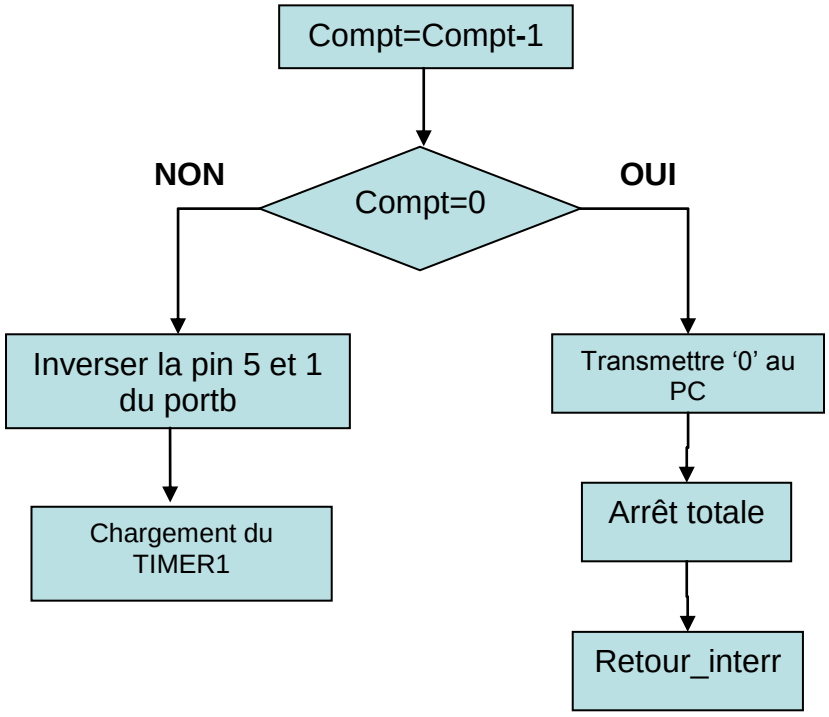

Figure.IV. 12: L'organigramme Gestion de l'interruption déclenchée par le Timer1 dans le mode de commande par trajectoire

IV.1.5.L'interface :

Pour faciliter la commande de notre robot nous avons réalisé une interface avec le logiciel **Borland C++ Builder** qui est un environnement de programmation visuel.

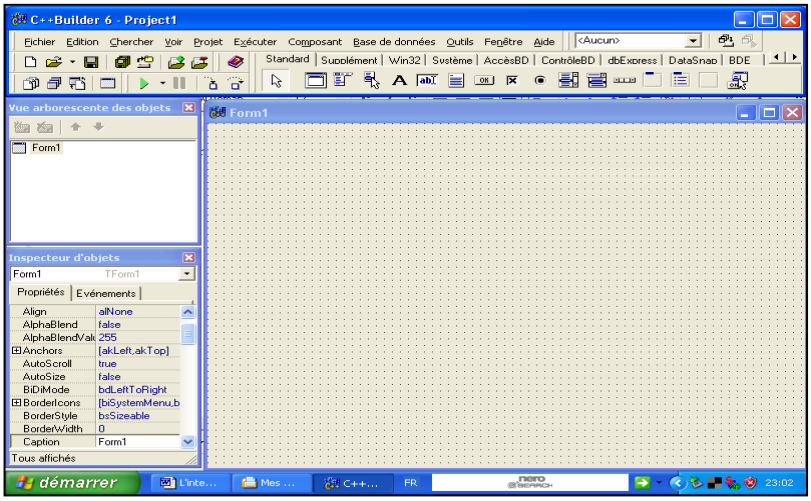

Figure IV.13: Borland C++ Builder

Nous avons réalisé 3 interfaces :

IV.1.5.1.L'interface principale :

Cette interface nous permet de choisir le mode de commande soit console de commande ou commande trajectoire.

Figure IV.14: L'interface principale

IV.1.5.1.1.Console de commande :

La figure ci dessus représente l'interface de console de commande

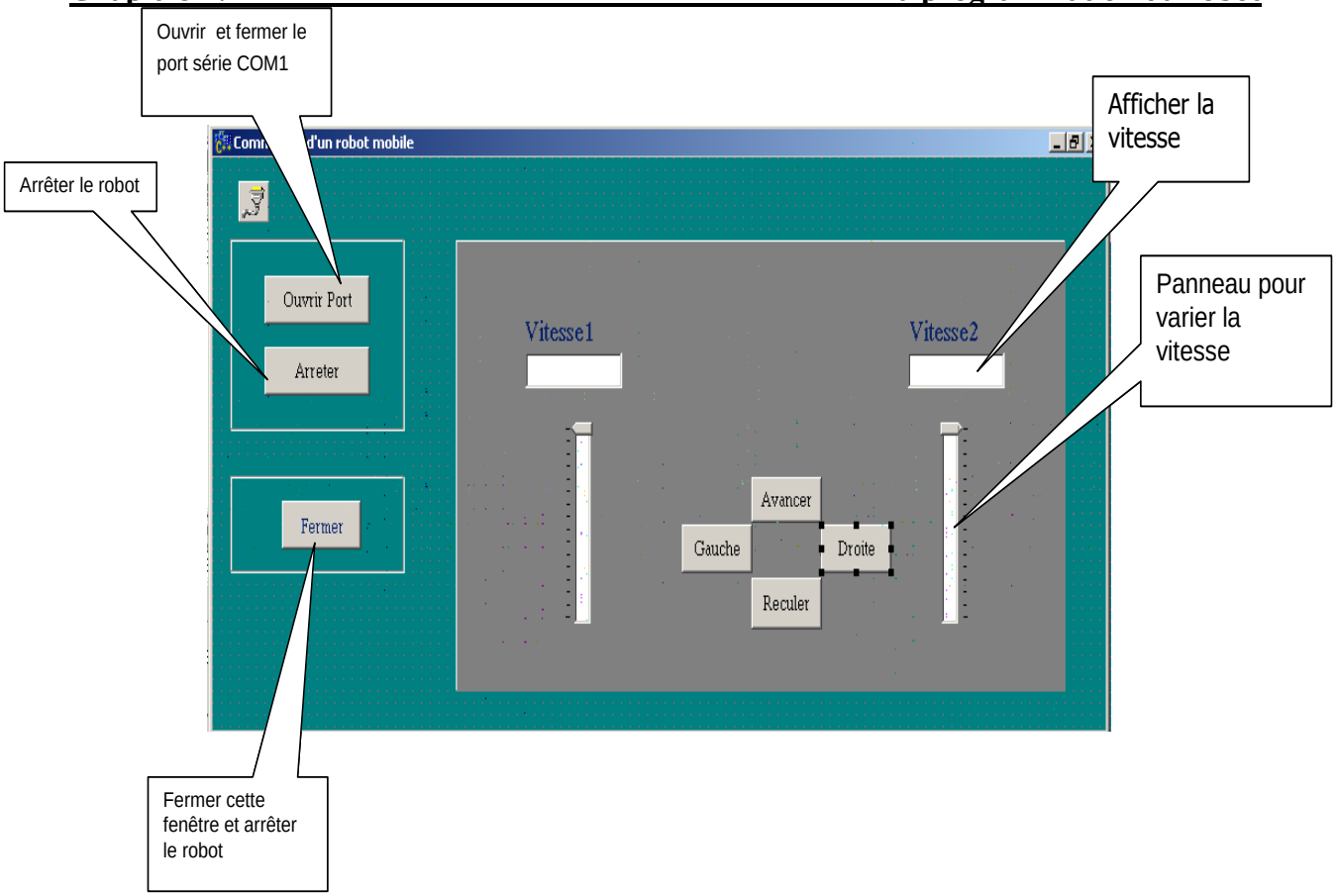

Figure IV.15: L'interface de console de commande

IV.1.5.1.2.Commande trajectoire :

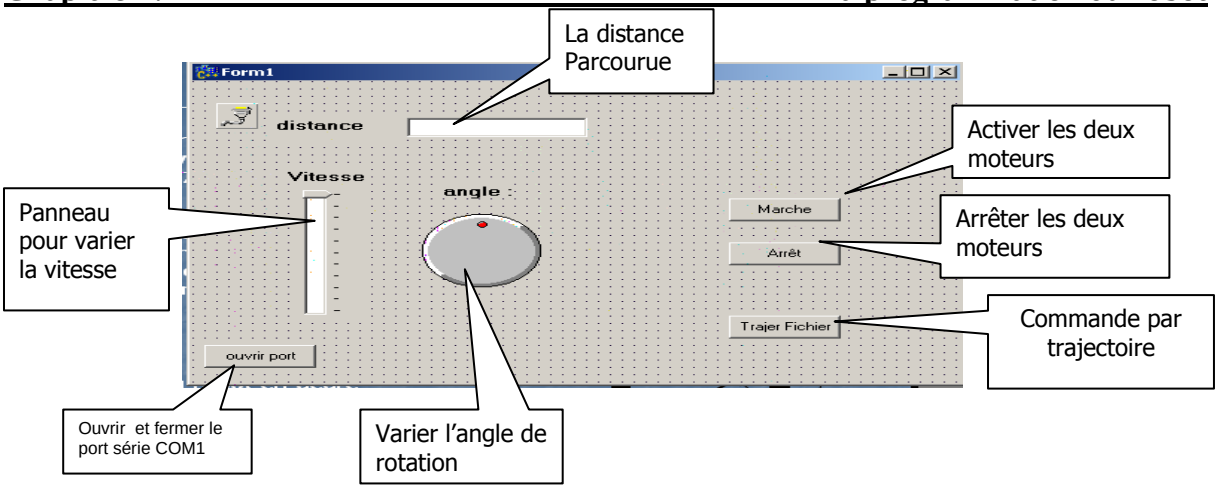

Figure IV.16 : Interface de commande trajectoire

L'organigramme suivant explique le fonctionnement de cette interface :

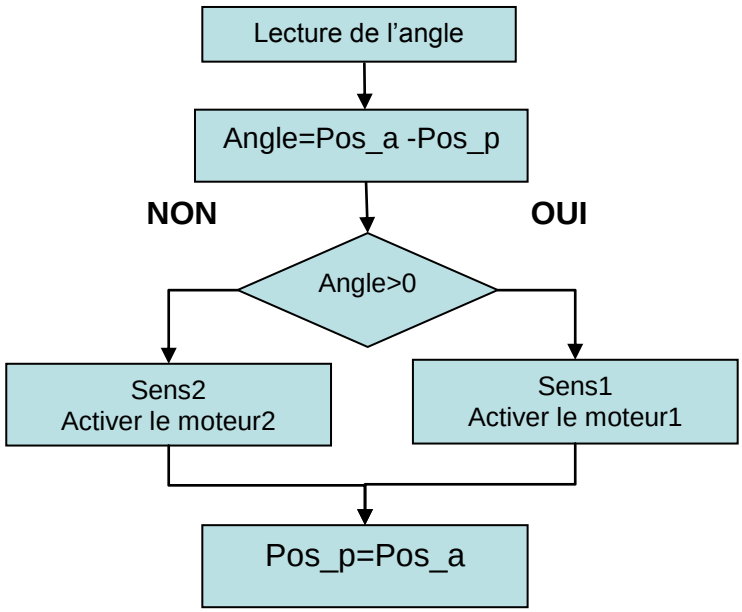

Figure IV.17 : L'organigramme de commande

Le principe consiste à envoyer au robot les coordonnées de la trajectoire à effectuer.

Ces coordonnées représentent le sens de rotation des deux moteurs et le nombre de pas à effectuer pour chaque partie de la trajectoire.

1-La trajectoire est une ligne droite :

Dans nos moteurs chaque pas correspond à 7.5°, et avec un rayon de la roue r =3.75cm, donc chaque pas provoque un déplacement de 'L'avec :
L = 2*π*r /48 = 0.4906 cm
Pour déterminer le nombre de pas afin que le robot parcourue une distance X :

$$N=X / 0.4906 \quad \text{------1}$$

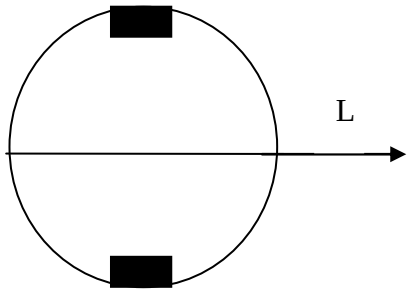

Figure IV.18: Trajectoire droite

2-La trajectoire est une rotation:

Dans cette étude on cherche le nombre de pas pour un angle de rotation du robot.
Les moteurs que nous avons utilisés font 7.5° (0.13 rad) par pas (moteur 48pas).
Le rayon de la roue r=3.75 cm donc le déplacement 'a' de la roue est obtenue par la relation suivante : a= 0.13*r

a=0.49cm => chaque pas, la roue se déplace par 0.49cm.

Nous avons le rayon de notre châssis R=12.5 cm.
La variable 'ß' est l'angle de rotation du robot, donc :
ß (rad)= a/R =0.03925 rad = 2.25°

Enfin pour un pas le robot tourne avec un angle de 2.25°
La formule générale pour obtenir le nombre de pas pour que le robot fasse un angle, est la suivante :

$$N= \Omega /2.25° \quad \text{------2}$$

N : Nombre de pas
Ω : L'angle de rotation du robot en degré

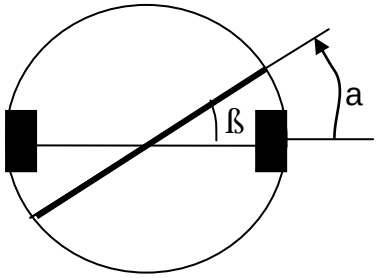

Figure IV.19: Trajectoire est une dérivation

3-Exemple d'une trajectoire :

A-B=L= 1m
B-C=0.5m
C-D=0.5m
Ω=30°
ß=100°

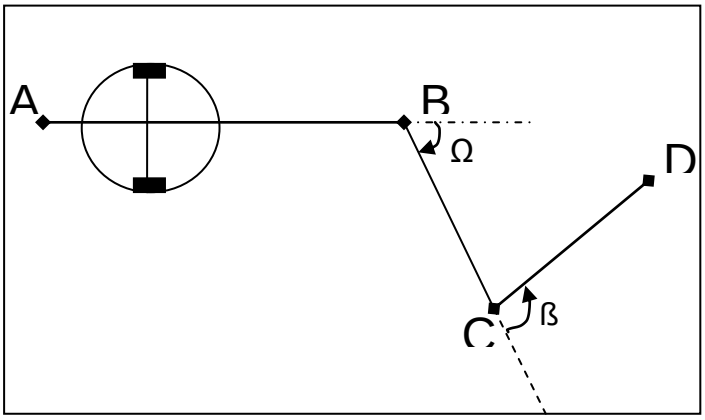

Figure IV.20: Exemple d'une trajectoire du robot

On obtient le nombre de pas en fonction de la trajectoire parcourue par le robot on appliquant la relation 1
Pour le cas d'une trajectoire droite :
N=L (cm) / 0.4906.
On applique la relation 2 pour une rotation avec un angle :
N= Ω/2.25°

Pour A-B :

N=100/0.4906=204 pas.
Au point B le robot s'arrêt et attend la nouvelle commande.

Le robot tourne avec un angle de 30° dans le sens négatif :

N= Ω /2.25°=30°/2.25°
N=13 pas
Le robot fait 13 pas et il s'arrêt jusqu'à la nouvelle commande.

Pour B-C :

N=50 /0.4906=102 pas
Au point C le robot s'arrêt et attend la nouvelle commande.

Le robot tourne avec un angle de 100 ° dans le sens positif :

On applique la relation 2 :
N= ß /2.25°=100°/2.25°
N=44 pas
Le robot fait 44 pas et il s'arrêt et attend la nouvelle commande.

Pour C-D :

N=50/0.4906=102 pas
Au point D le robot s'arrêt.

L'appel de ce fichier texte permet au robot de faire cette trajectoire ;

```
//Trajectoire A-->B
 VaComm1->WriteChar (vitesse1+64);
 VaComm1->WriteChar (0xCC);
 VaComm1->WriteChar (0x00);
 VaComm1->WriteChar (0x02);
//Angle Ω
 VaComm1->WriteChar (vitesse1+64+32);


 VaComm1->WriteChar (0x0D);
 VaComm1->WriteChar (0x00);
 VaComm1->WriteChar (0x02);
//Trajectoire B-->C
 VaComm1->WriteChar (vitesse1+64);
 VaComm1->WriteChar (0x66);
 VaComm1->WriteChar (0x00);
 VaComm1->WriteChar (0x02);
 //Angle ß
 VaComm1->WriteChar (vitesse1+64+16);
 VaComm1->WriteChar (0x2C);
 VaComm1->WriteChar (0x00);
 VaComm1->WriteChar (0x02);
//Trajectoire C-->D
 VaComm1->WriteChar (vitesse1+64);
 VaComm1->WriteChar (0x66);
 VaComm1->WriteChar (0x00);
 VaComm1->WriteChar (0x02);
```

IV.6 .Conclusion:

Dans ce chapitre nous avons présenté les différents algorithmes du programme implémenté dans le PIC du robot, et nous avons aussi présenté les différents modes de commande du robot ainsi l'interface qui facilitent la commande du robot.