Chapitre 4

Manuel d'utilisation et simulation

4.1 Manuel d'utilisation de l'émulateur

La Figure 4.1 représenter l'interface graphique de l'émulateur didactique de fonctionnement de microprocesseur 8086, il contient 9 parties :

1. Le menu principal.
2. Le panneau de control.
3. La zone d'affichage de code.
4. La zone d'affichage des registres.
5. Lazone d'affichage des variables et constants.
6. La zone d'affichage des tableaux.
7. Lazone d'affichage d'Entrées/Sorties.
8. la zone d'affichage de la pile.
9. Lazoned'affichage des informations sur la compilation et lasyntaxe de code.

L'application de la Figure 4.1 est chevronnée d'exécuter 49 type d'instructions similaire au jeu d'instructions de microprocesseur 8086, l'application est simple a utilisé car les zones d'affichage sont fixées afin de bien présenter les changements des informations concerné l'exécution de programme assembleur charger.

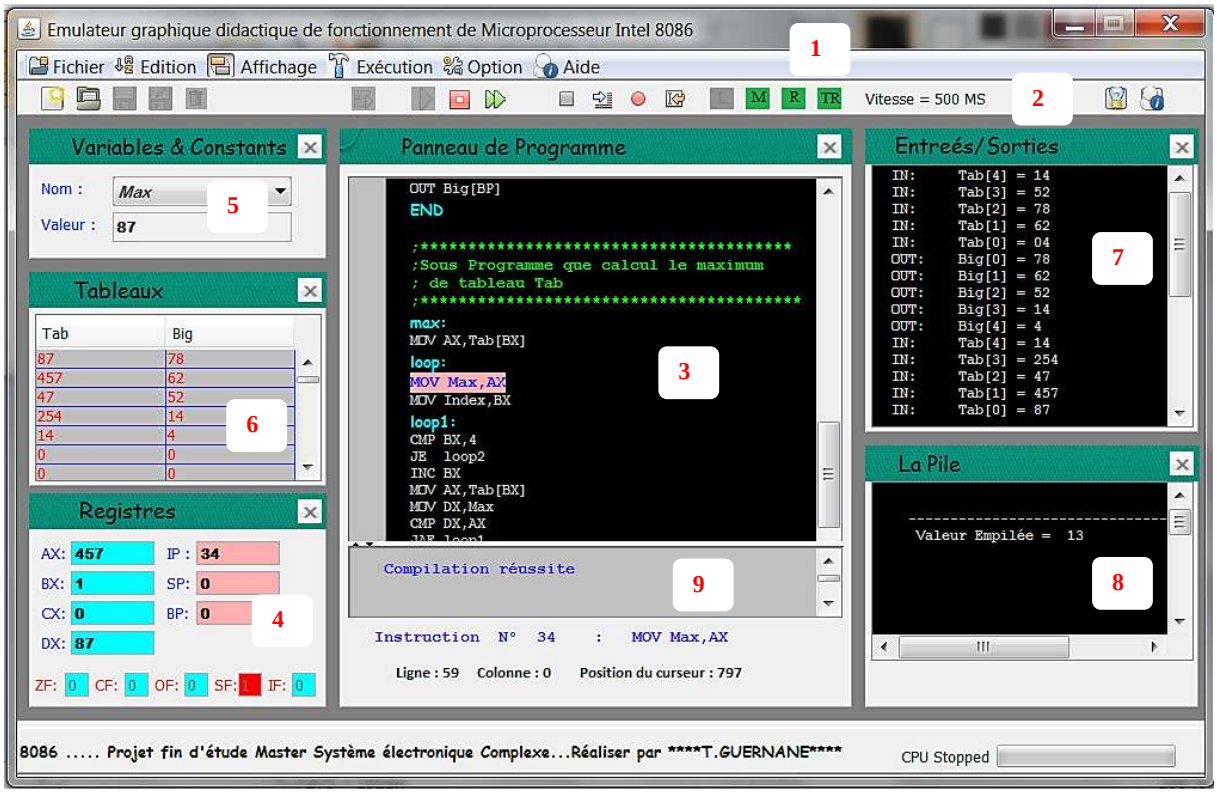
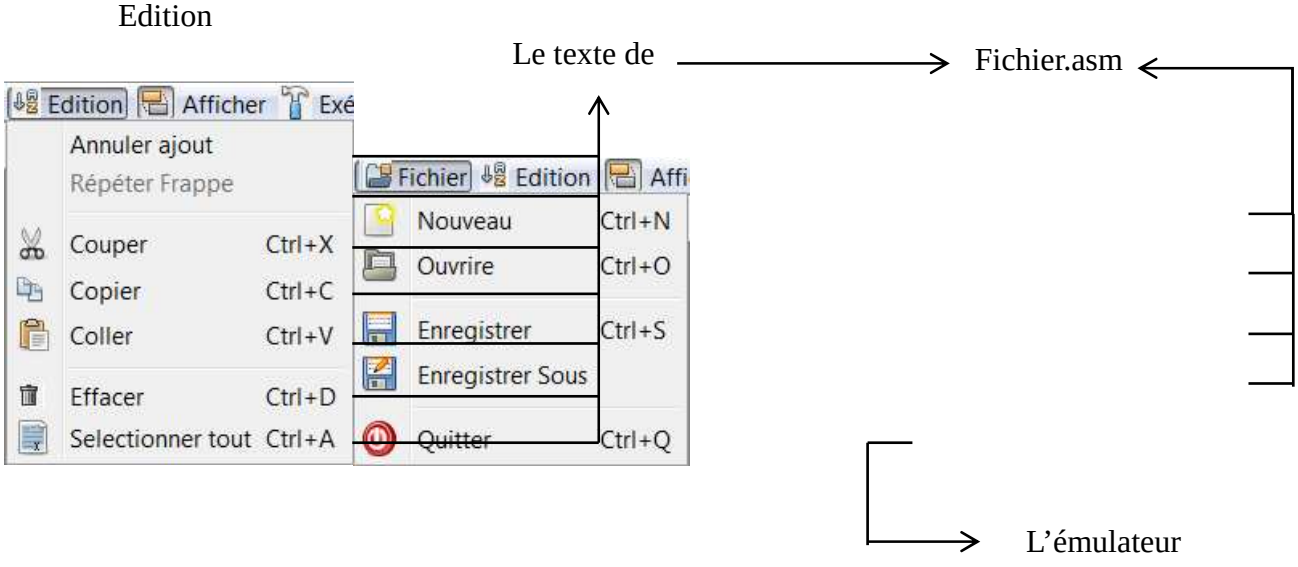
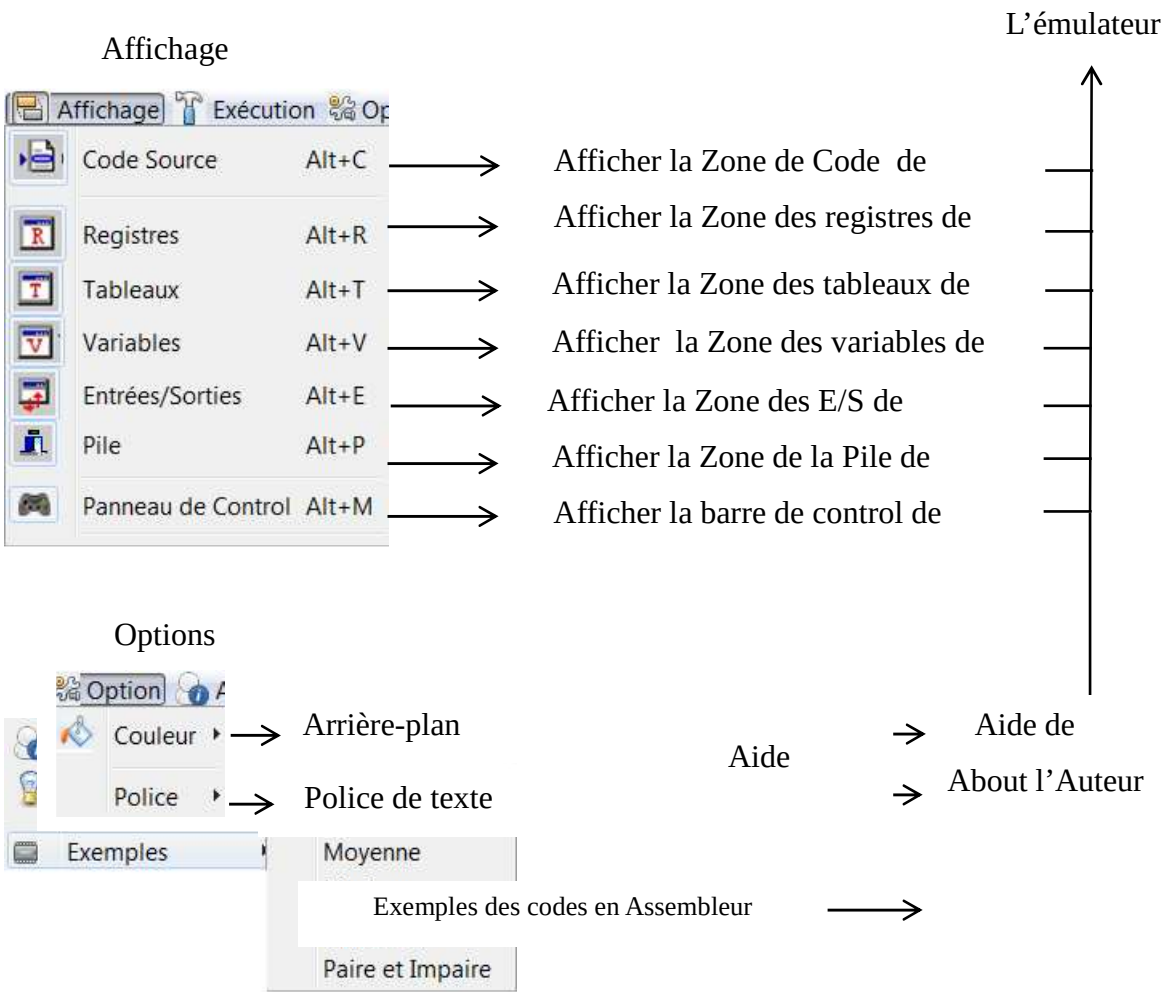


Fig.4.1 / L'interface graphique de l'émulateur.

4.1.1 Le menu principal de l'émulateur

Le menu principal de l'émulateursur contient 6 titres chaque un de ces titres contient des sous-titres que spécifions a l émulateur une tâche à réaliser.Fig.4.2.





Exécution de programme

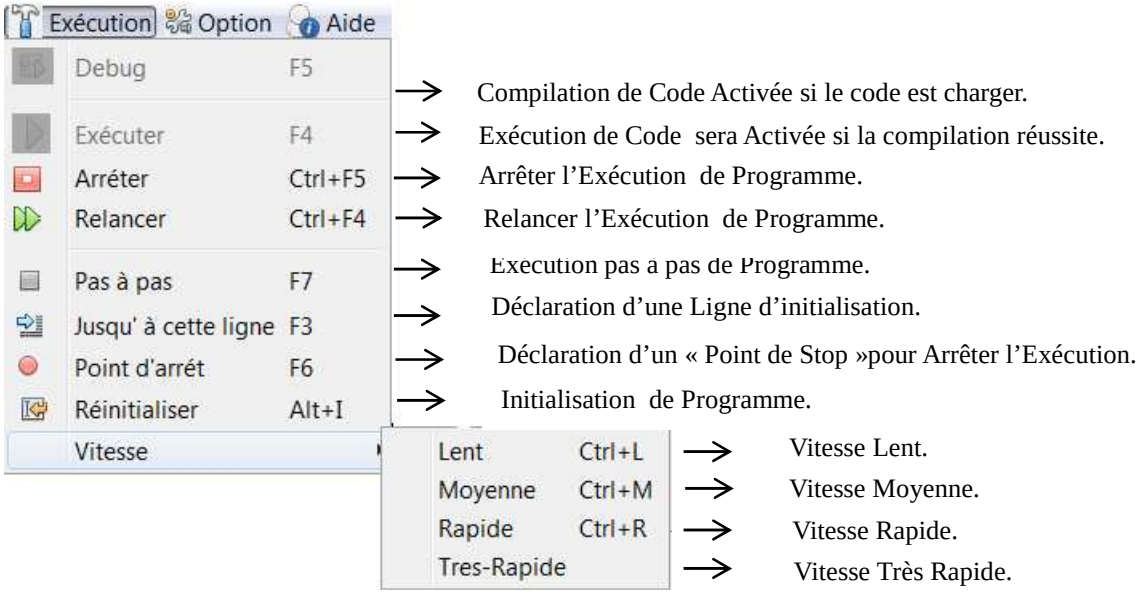


Fig.4.2 / le menu principal de L'interface graphique.

4.1.2 Lepanneau de contrôle de l'émulateur

Icône 	Ecrire un nouveau Code.
Icône 	Charger un nouveau Code.
Icône 	Enregistrer le Code.
Icône 	Enregistrer sous le Code.
Icône 	Effacer le Code.
Icône 	Compilation de Programme.
Icône 	Lancer l'exécution de Programme.
Icône 	Arrêter l'exécution de Programme.
Icône 	Relancer l'exécution de Programme.
Icône 	l'exécution pas à pas de Programme activée si la simulation est en halte.
Icône 	Déclaration de la ligne d'Initialisation Programme.
Icône 	Déclaration du point de d'arrêt de Programme.
Icône 	Initialisation de Programme en a n'importe quel moment.
Icône 	L'exécution de programme avec une vitesse retardée de 500 Ms.
Icône 	L'exécution de programme avec une vitesse retardéede 250 Ms.
Icône 	L'exécution de programme avec une vitesse retardée de 50 Ms.
Icône 	L'exécution de programme avec une vitesse retardée de 10 Ms.

Tab.4.1/ panneau de control de l'interface graphique de l'émulateur.

4.2 Première utilisation de l'émulateur

L'utilisation de l'émulateur est très simple, Il se fait de charger un nouveau code ou taper le code sur l'Editor de texte, le code doit avoir une extension « .asm », si il n'y a pas des erreurs de syntaxe le bouton de compilation est alors activée l'utilisateur peut compiler le code si la compilation est réussite, l'interface graphique mise à jour tous les données et les registres ainsi l'activation les boutons de simulation, et les zone d'affichage. si non l'utilisateur doit corriger le code saisi.

4.2.1 Chargement de code Assembleur

Si l'utilisateur click sur le bouton Ouvrir, la fenêtre de dialogue Fig.4.3 apparaitra, l'utilisateur peut charger un nouveau code, ainsi la fenêtre afficher que les fichiers en Assembleur afin de simplifier le chargement de code source.

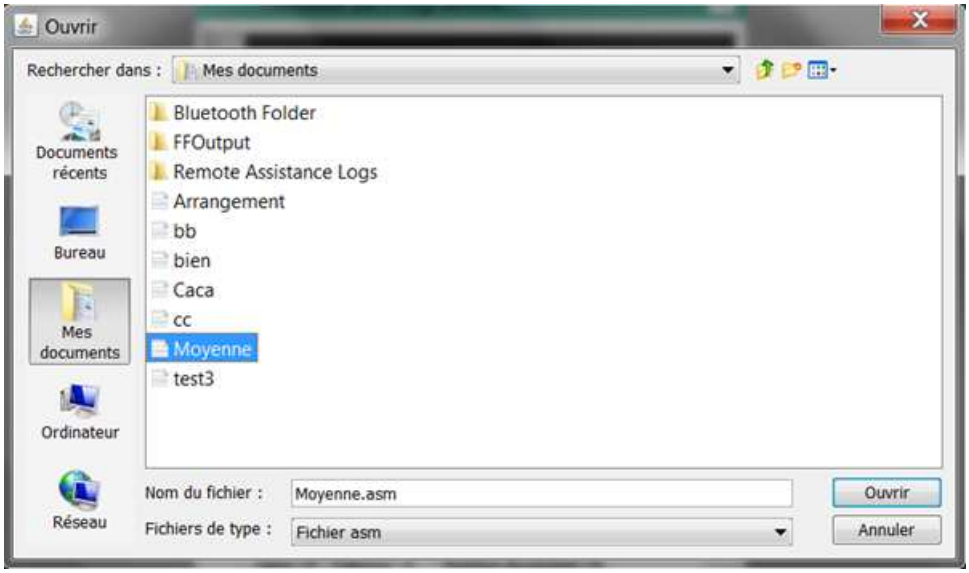


Fig.4.3Charger un nouveau code d'Assembleur.

Si le code est chargé et s'il n'existe pas des erreurs de syntaxe, il soit affiché dans la zone de code comme indique la figure Fig.4.4. Ainsi le bouton de compilation est activée automatiquement pour qui l'utilisateur commencer la compilation le code.

Nous avons choisi un exemple d'un programme que calcule la somme des valeurs saisi d'un tableau de taille 5 et calculé la moyenne des ces valeur est afficher les résultants de calcul la figure4.4 Indique la syntaxe de code charger il contient deux zone la zone des données commence par le Mot-clé « DATA », et la zone du code fonctionnelle commencer par le Mot-clé « CODE », la zone code doit commencer par une étiquette « start » et terminer par l'indicateur de la fin de code Assembleur nommé « AND ».

```
;Programme Calcul de moyenne
DATA
somme   DW   0
moyenne   DW 0
maximum   DW 5
notes TB 5
CODE
start:
MOV AX,0
MOV BX,0
loop:
IN notes[BX]
ADD AX,notes[BX]
INC    BX
CMP BX,maximum
JNE  loop
MOV somme ,AX
DIV BX
MOV moyenne ,AX
CALL affichage
END
affichage :
OUT somme
OUT moyenne
```

Fig.4.4 Exemple d'un Code d'assembleur Chargé.

4.2.2 Syntaxe et Compilation de code Assembleur

- **Syntaxe de code Assembleur**

S'il existe une erreur de syntaxe leur de chargement d'un code l'erreur sera affichée sur la zone de « compilation et de syntaxe » de l'émulateur comme indique la sur figure 4.5.

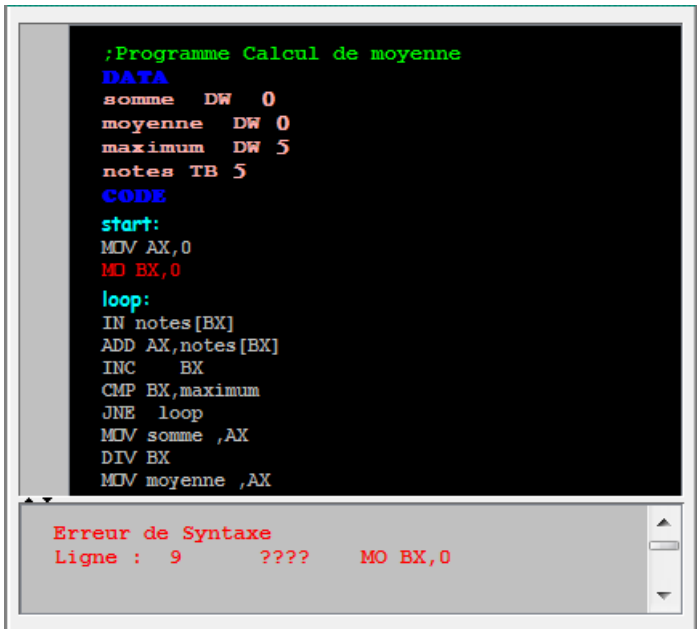


Fig.4.5 / Syntaxe de Code Assembleur chargé.

- **Compilation de code Assembleur**

Si le code est erroné le message indique sur la figure 4.6.B sera afficher, afin d'indique un échec de la compilation de code Assembleur charger. Par contre si le code est correcte le message de la figure 4.6.A sera afficher pour indique la réussite de compilation de code.

A **B**

Fig.4.6 / Confirmation de la Compilation de Code Assembleur Chargé.

4.2.3 Première simulation

La figure 4.7 indique une première simulation de programme de calcul de moyenne d'un tableau « notes » de taille 5, les résultants des calculs de la « somme » et la moyenne sontafficher sur la zone d'entrées/sorties.

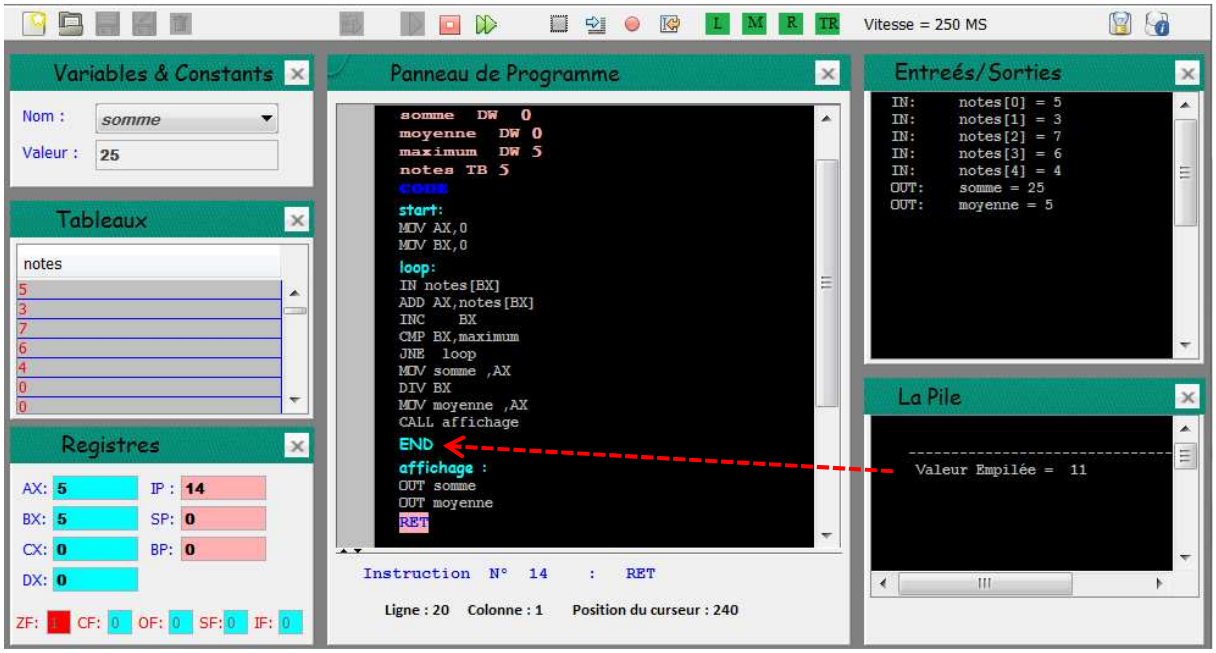


Fig.4.7 / première Simulation de Code AssembleurChargé.

4.3 Analyse de la simulation

4.3.1 Analyse des données

La figure 4.8 indique les données de programme simulé à la fin d'un cycle d'exécution.

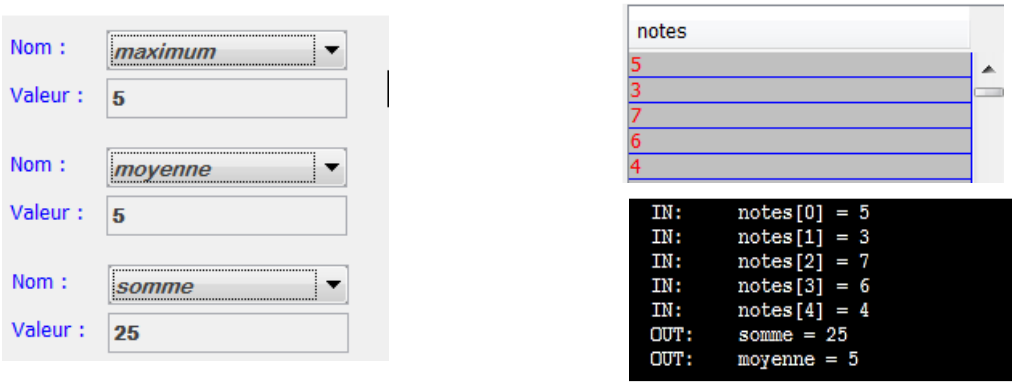


Fig.4.8 / Analyse les données du Code Chargé.

4.3.2 Analyse des Registres

La figure 4.9 indique la mise à jour des registres interne de l'émulateur à la fin d'un cycle d'exécution de programme simulé. On voit que le registre de travail AX contient la valeur de la moyenne = 5 et le registre BX contient la valeur maximum de la taille du tableau « notes ».

La valeur de pointeur d'instruction IP = 14, il est pointé sur l'instruction « RET » afin desortir de sous-programme d'affichage.La valeur de IP avant l'appelle a ce sous-programme d'affichage est empilées dans la pile, ou IP = 11 c'est l'adresse de la dernier instruction a décodé « END ».

Le flag ZF est positionné a 1 car à la fin d'exécution de l'instruction « CMP BX, maximum » la soustraction de BX de 5 a donné 0, le programme acontinué l'exécution et il ne fait un saut vers l'étiquette « loop » car le test effectué par l'instruction « JNE » a échoué

Fig.4.9 / Analyse les registres interne de l'émulateur.

4.3.3 Analyse de la vitesse de simulation

L'émulateur il peut travail avec quatre vitesses simulation de programme en réalité ces sont pas des vitesses d'exécution mais des retards car la vitesse d'exécution est la vitesse de la machine virtuel de java « JVM », nous avons retardé l'exécution de programme Assembleur simulé comme indique la figure 4.9.

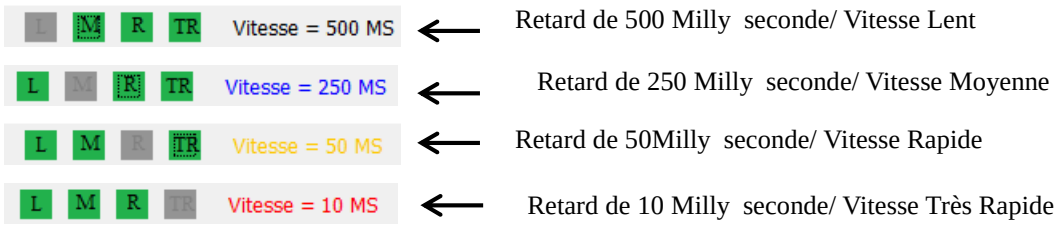


Fig.4.10 / Analyse de la vitesse de programme Assembleur simulé.

4.3.4 Analyse de la pile

La figure 4.11 indique la valeur de pointeur d'instruction IP empilée lorsque l'instruction « CALL affichage » appelle le sous-programme d'affichage des résultants du calcul du moyenne. La valeur du SP est incrémentée de « -1 » à « 0 » comme indique sur la figure 4.9.

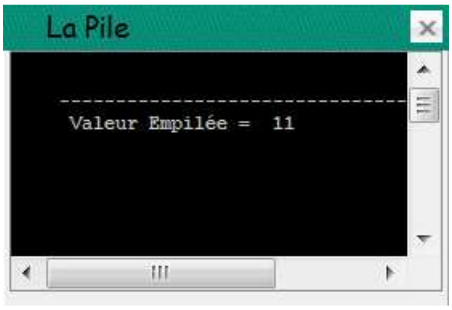


Fig.4.11 / Analysé la pile de programme Assembleur simulé.

4.3.5 Analyse les entrées/sorties et les interruptions

Afin d'explique le mécanisme d'utilisation des vecteurs d'interruption dans un programme Assembleur simulé sur l'émulateur nous avons choisi l'exemple de la figure 4.12 que indique l'utilisation de 5 vecteur d'interruption chaque vecteur lire un caractère du la mémoire ROM qui contient 65 caractères puis il affiche ce caractère sur la zone d'entries/sorties.

Le programme de la figure 4.12 chargé dans « AX, BX, CX, DX, CX, BP » respectivement les numéros des caractères à lire suivants « 12, 4, 17, 2,8 » pour qui a la fin le mot affiché soit « MERCI »

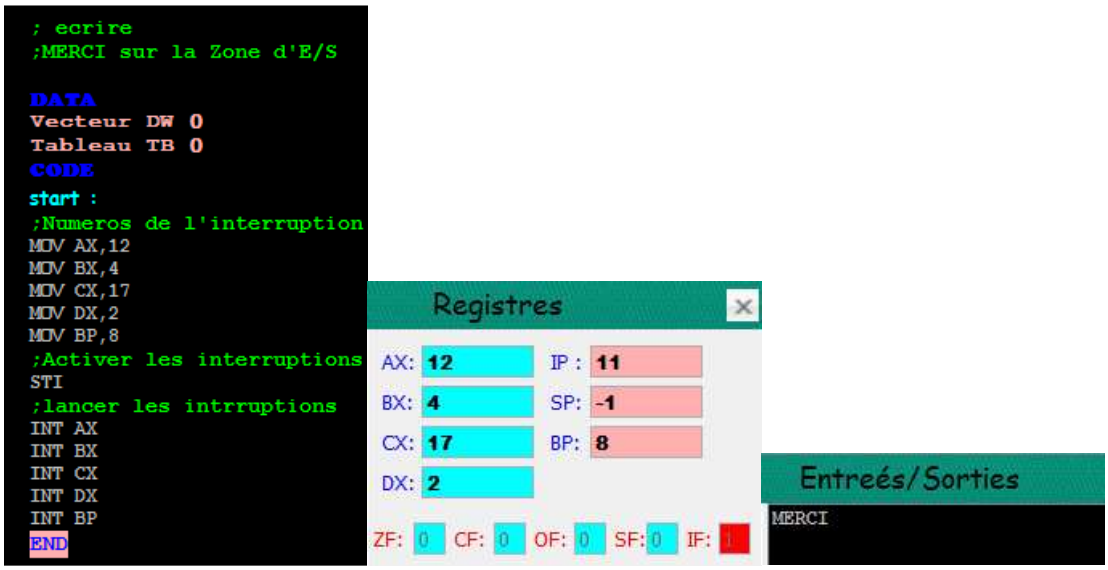


Fig.4.12 / Utilisation des interruptions.

4.4 Exemplesd'application

Pour plus de détail etpour clarifie le travail réaliser voici quelque exemples des codes en assembleur qui sont était testé sur l'émulateur

4.4.1 Calcul le nombre maximum des valeurs d'un tableau

Le programme suivant calcul le maximum des valeurs d'un tableau de taille cinq, l'émulateur lire les cinq entiers positive puis il traité ces données afin de calcul la valeur Max entre aux.

```
; Calcul le  maximum des valeurs d'un tableau
            DATA
            Max DW 0
            Tab TB 5
            CODE
            start:
            MOV AX,0
            MOV BX,4
            IN Tab[BX]
            DEC BX
            IN Tab[BX]
            DEC BX
            IN Tab[BX]
            DEC BX
            IN Tab[BX]
            DEC BX
            IN Tab[BX]
            MOV AX,Tab[BX]
; Charger la valeur maximum dans Max
            max:
            MOV Max,AX
;Test si la valeur est Max
            loop:
            CMP BX,4
            JE  afficher
            INC BX
            MOV AX,Tab[BX]
            MOV DX,Max
            CMP DX,AX
            JAE loop
            JBE max
;Affichage de valeur Max
            afficher:
            OUT Max
            END
```

Simulation de Programme :

Les résultants de simulation de premier exemple sont montrés sur la figure4.14.

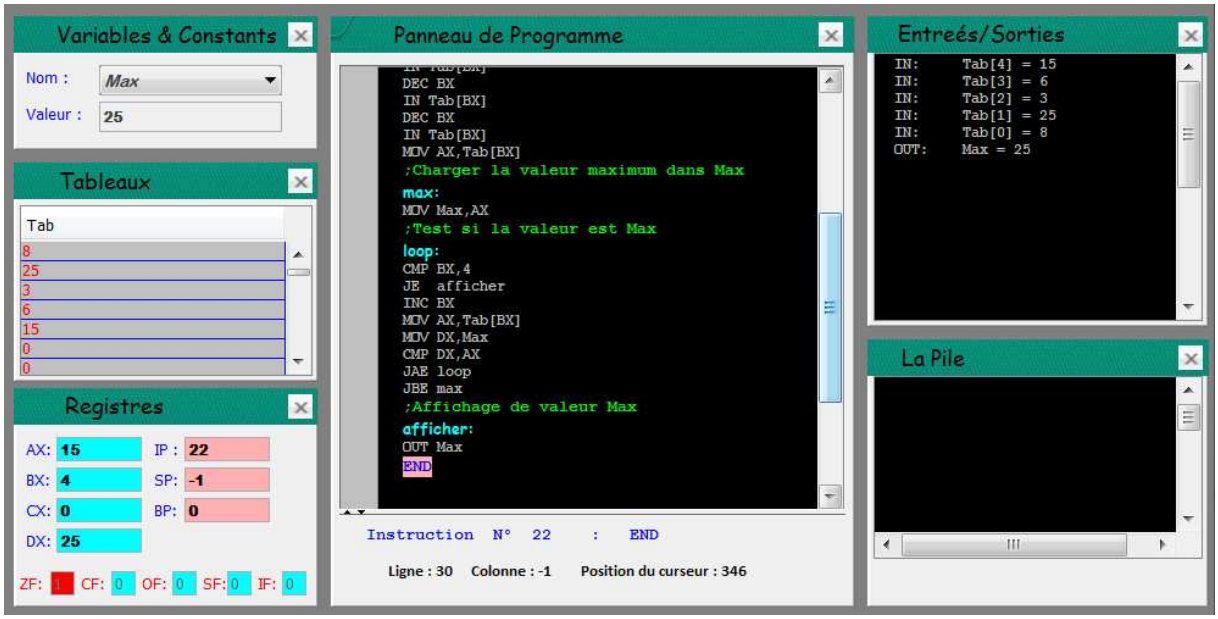


Fig.4.13 Simulation de premier exemple.

La figure 4.14 indique les données de programme a la fin de simulation, on a entré 5 valeurs de 0 a 5 comme suite « 8, 25, 3, 6, 15 », la valeur Max des ces nombres est était 25.

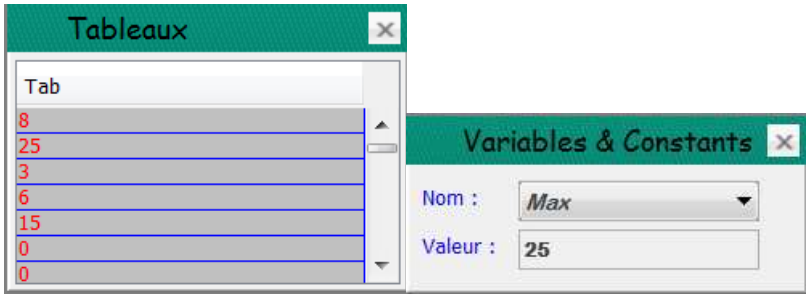


Fig.4.14Les données de premier exemple.

La figure 4.15 indique l'affichage de la valeur Max entre les cinq valeurs entrées, la valeur afficher est 25 car c'est la valeur Max.

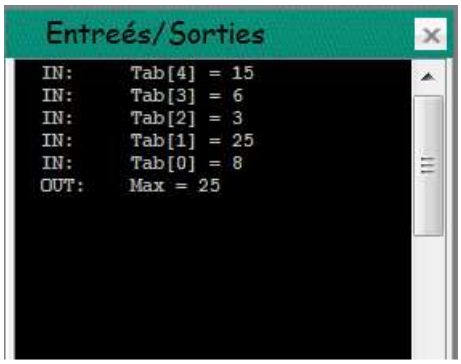


Fig.4.15Entrées/Sorties de premier exemple.

Fig.4.16Les registres interne de premier exemple

La figure 4.16 indique les registres internes de l'émulateur a la fin de la simulation en a travaillé avec 3 registres AX et DX pour effectue des calculs, et BX pour l'adressage des cases de tableau « Tab » ou AX il contient la valeur d'un case a test si cette valeur et supérieure a la valeur de DX le variable Max est charger si non le test continue jusque la dernière valeur de tableau BX= 4 a cette valeur le test est terminer le programme afficher la valeur Max en Sortie.

4.4.2 Ordonner les valeurs d'un tableau dans un autre

```
;Ordonner les Valeurs d'un tableau dans un autre
            DATA
            Max DW 0
            Index DW 0
            Tab TB 5
            Ordonner TB 5
            CODE
;Lecteur d'un Tableau de taille 5
            start:
            MOV BX,4
            IN Tab[BX]
            DEC BX
            IN Tab[BX]
            DEC BX
            IN Tab[BX]
            DEC BX
            IN Tab[BX]
            DEC BX
            IN Tab[BX]
;Programme Principal
            label:
            CMP BP,5
            JAE affichage
            CALL max
            MOV BX,Index
            MOV AX,0
            MOV Tab[BX],AX
            MOV AX,Max
            MOV Ordonner[BP],AX
            INC BP
            MOV AX,0
            MOV BX,0
            JMP label
;affichage des valeurs Ordonner
            affichage:
            MOV BP,0
            OUT Ordonner[BP]
            INC BP
            OUT Ordonner[BP]
            INC BP
            OUT Ordonner[BP]
            INC BP
            OUT Ordonner[BP]
            INC BP
            OUT Ordonner[BP]
            END
;Sous-programme de calcul le maximum
            max:
            MOV AX,Tab[BX]
            loop0:
            MOV Max,AX
            MOV Index,BX
            loop1:
            CMP BX,4
            JE  sortie
            INC BX
            MOV AX,Tab[BX]
            MOV DX,Max
            CMP DX,AX
            JAE loop1
            JBE loop0
            sortie:
            RET
```

Le programme ci-dessus ordonner les valeurs d'un tableau de taille 5 d'une façon descendante dans un autre tableau de la même taille, l'ascien tableau sera vidé.

Simulation de deuxième exemple :

Les résultants de simulation sont montrés sur la figure 4.17.

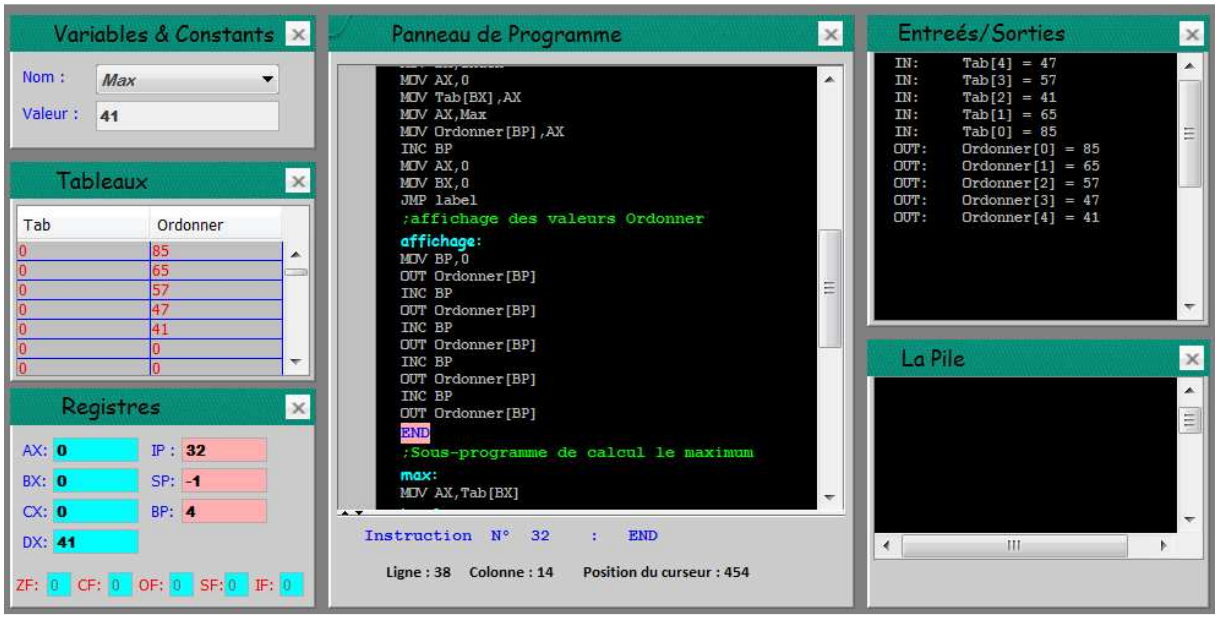


Fig.4.17 Simulation de deuxième exemple.

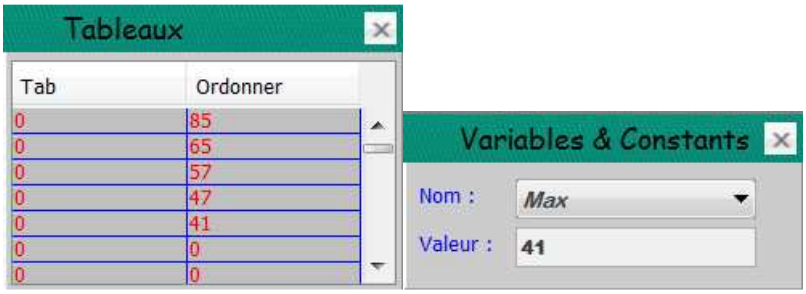


Fig.4.18Les données de deuxième exemple.

La figure 4.18 indique les données de programme a la fin de simulation on observe que le tableau « Tab » est vidé et le nouveau tableau « Ordonner » est chargé de façon descendante

par les valeur de « Tab » la dernière valeur charger sur le variable Max est 41 car c'est la valeur Minimum de tableau « Tab ».

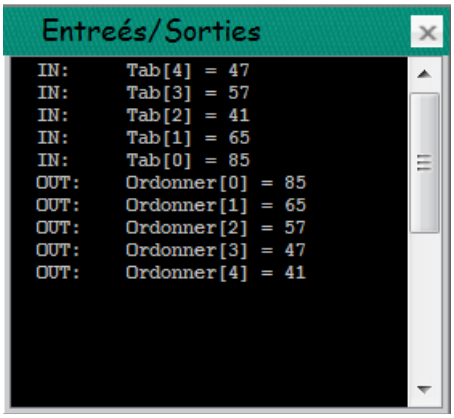


Fig.4.19Entrées/Sorties de deuxième exemple.

La figure 4.19 indique l'entrées/sorties de deuxième programme, on a entrées les valeurs de 4 a 0 dans le tableau « Tab » comme suit (47, 57, 41, 65, 85) le programme ordonner ces valeur dans le tableau « Ordonner » de 0 a 4 comme suit (85, 65, 57, 47, 41) donc d'une façon descendante.

Fig.4.20Registres internes de deuxième Exemple.