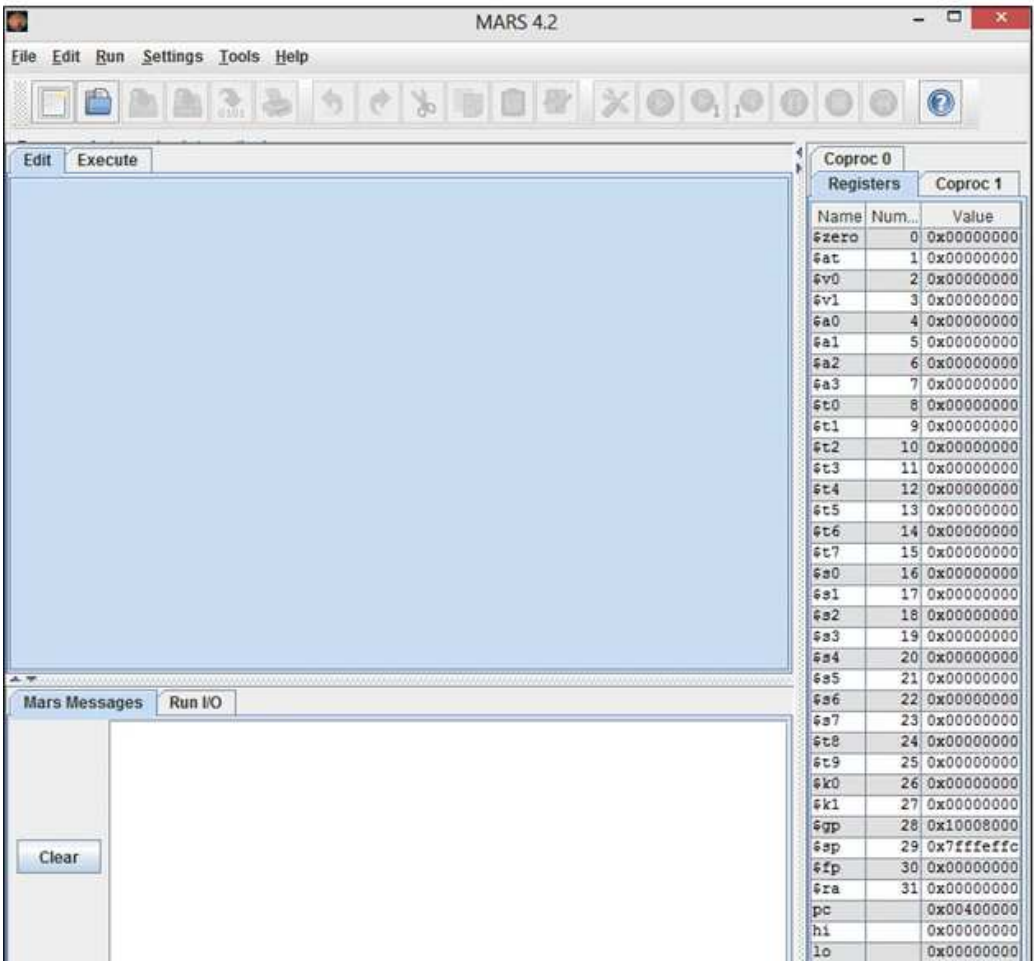
# TP N°01

## *Partie 1 : Interface de l'IDE Mars (MIPS Assembler and Runtime Simulator)*

Le Mars 4.5 est un simulateur du langage d'assemblage du processeur MIPS. Il se présente sous la forme d'un fichier « .jar » exécutable avec la JRE de Java installée au préalable sur la machine. On trouve au centre de l'interface les onglets « Edit » pour l'édition des programmes et « Execute » pour visualiser le résultat de l'assemblage. En bas, nous pouvons visualiser les messages du simulateur et les résultats des entrées/sorties grâce aux onglets « Mars Messages » et « Run I/O »

A droite, trois onglets permettent de visualiser les registres du processeur : Les 32 registres généraux plus d'autres registres importants, les 32 registres f0-f31 du Coprocesseur0 pour les opérations de multiplication et de division, et les registres protégés du processeur dans le Coprocesseur1.




Pour éditer un programme en langage d'assemblage : Menu principal → File→New, on édite le programme dans l'onglet « Edit ». On enregistre ensuite le programme avec l'extension « .asm » ou « .s ».

On compile ou assemble le programme : Menu principal → Run → Assemble, et on exécute le programme Menu principal → Run → Go

## *Partie 2 : Exercices*

### *Exercice 0 :*

1- Taper le code suivant, et l'enregistrer sous le nom « prog0.asm » :

```
.data
        coef:   .byte   3
.text
```

Assembler le programme, et observer les trois panneaux « text segment », « data segment » et « labels » dans l'onglet « Execute ». Si le panneau « Labels » n'apparait pas, on peut l'afficher via : Settings→ Show Labels window (Symbol table). Si il y'a des erreurs, elles s'afficheront en bas dans l'onglet « Mars Messages ».

Effectuer maintenant l'exécution du programme, et voir le résultat de l'affichage dans l'onglet «Run I/O ».

2- Modifier le code en ajoutant la déclaration de deux variables supplémentaires coefd et pid :

```
.data
        coef:       .byte       3
        coefd:      .float      6
        pid :       .double     6.28
.text
```

Assembler le programme une nouvelle fois, et observer les trois panneaux « text segment », « data segment » et « labels » dans l'onglet « Execute ».

3- Ajouter maintenant une instruction dans la section de texte après la directive « .text » :

```
.data
        coef:       .byte       3
        coefd:      .float      6
        pid :       .double  6.28
.text
        addi $t0,$t0,5
```

Assembler le programme une nouvelle fois, et observer les trois panneaux « text segment », « data segment » et « labels » dans l'onglet « Execute ». Observer les valeurs des registres $t0 et pc. Exécuter le code et observer une nouvelle fois les valeurs des registres $t0 et pc.

4- Modifier l'instruction « addi $t0, $t0, 5 » en « addi pc, pc, 5 » et assembler. Que dit le résultat de l'assemblage ?

### *Exercice 1 :*

Ecrire et exécuter le programme « Hello World » (support TP). Le programme doit être sauvegardé sous le nom « helloworld.asm ». A l'assemblage, visualiser les panneaux « Text Segment » et « Labels » et observer les résultats des différents champs. Observer aussi le contenu des registres du processeur à droite, notamment les registres $a0 et $v0.

***Exercice 2 :*** Ecrire un programme en assembleur nommé "*Caractere*.asm " qui affiche un caractère à l'écran.

***Exercice 3 :*** Ecrire un programme en assembleur nommé "*Entier.asm*" qui affiche la valeur d'un entier à l'écran

**Remarque**

A l'assemblage si il y'a des erreurs, elles vont apparaitre en bas dans la partie « Mars Messages ». Bien lire les messages, corriger l'erreur à la ligne indiquée et refaire l'assemblage, ne pas déranger le chargé de TP pour rien !