

# TP 2 : Utilisation d'un simulateur de système d'exploitation

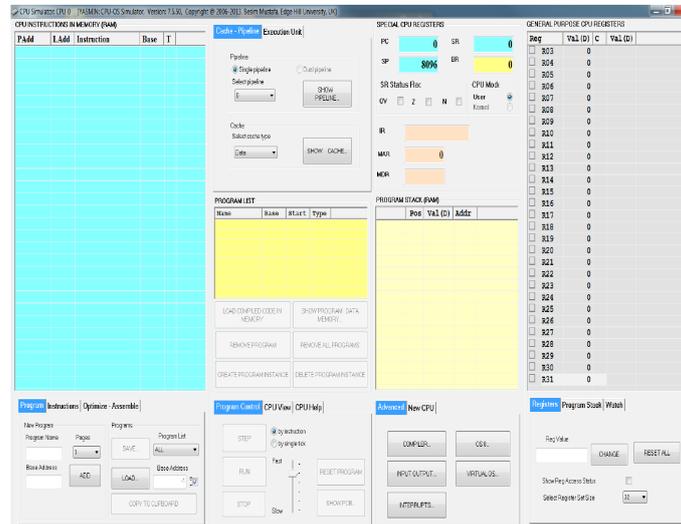
## NFP136 : Valeur d'Accueil et de Reconversion en Informatique 2

### Supports :

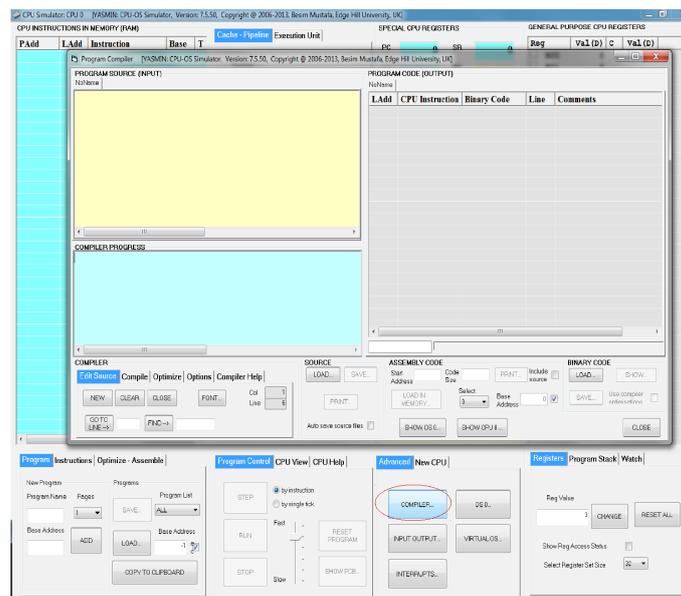
Tous les supports pour ce tp sont disponibles sur à l'adresse : <http://deptinfo.cnam.fr/new/spip.php?rubrique227>

Le but de ce tp est de manipuler un simulateur de processeur et de système d'exploitation. Ce simulateur permet de visualiser et d'animer les mécanismes mis en oeuvre dans le fonctionnement du système d'exploitation. Il permet d'observer les taches cachées du système d'exploitation.

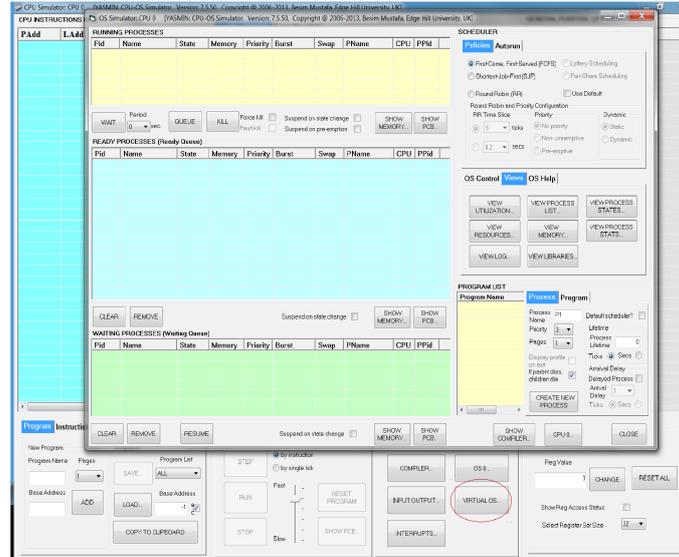
Lancer le simulateur, la fenêtre qui représente le CPU s'ouvre :



Vous pouvez accéder au compilateur via le bouton **COMPILER** :



Vous pouvez accéder au système d'exploitation virtuel via le bouton **VIRTUAL OS** :



Vous pouvez remarquer que sur les fenêtres **COMPILER** et **VIRTUAL OS**, il y a des boutons **SHOW OS**, **SHOW COMPILER** et **SHOW CPU** qui permettent la navigation d'une fenêtre à l'autre.

**Exercice 1 — La compilation** Le but de cet exercice est de vous illustrer les étapes de la compilation d'un programme. Pour rappel, un compilateur transforme un programme source en code objet (binary code) en trois étapes : l'analyse lexicale, l'analyse syntaxique et l'analyse sémantique. Au fil de ces trois étapes, le compilateur construit une table de symbole. Enfin, il génère le code objet. Dans cet exercice, nous ne considérerons pas d'édition de lien.

- 1) Dans la fenêtre CPU, cliquez sur le bouton **COMPILER** pour ouvrir la fenêtre **COMPILER**
- 2) Tapez dans le cadre **PROGRAM SOURCE (INPUT)** le programme suivant :

```

program EssaiCompilateur
  n=5
  for i = 1 to 3
    n = n + 1
    if n = 2 then
      n = 0
    end if
  next
end
  
```

Vous pouvez sauvegarder votre programme en cliquant sur le bouton **SAVE** du cadre **SOURCE**.

- 3) Compilez ce programme en cliquant sur le bouton **COMPILE** dans le cadre **COMPILER**, onglet **Compile**. Vous pouvez reconnaître le code machine, généré par la compilation, qui est affiché dans le cadre **PROGRAM CODE (OUTPUT)**. Plus précisément, vous pouvez voir la colonne **LAdr** qui est l'adresse logique de l'instruction, l'instruction dans la colonne **CPU Instruction**, le code binaire associé à l'instruction dans la colonne **Binary Code**, la ligne du code machine dans la colonne **line**, et dans la colonne **comments**, ce que fait l'instruction.



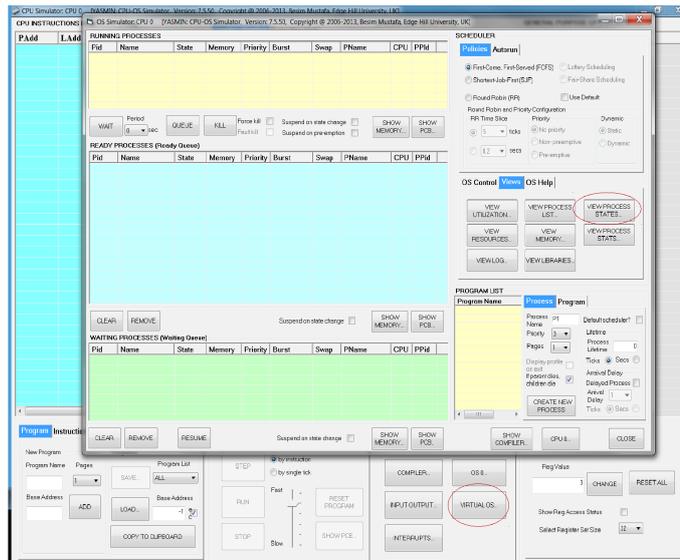
Observez ce qui se passe au niveau du processeur dans la fenêtre CPU du simulateur :

- a) Dans le cadre CPU INSTRUCTION IN MEMORY, vous pouvez voir l'état de la mémoire centrale (ou RAM) après chargement du programme. La première colonne PAdr est l'adresse physique de chaque ligne du programme, la deuxième LAdr leur adresse logique et la troisième Instruction est l'instruction en code machine.
- b) Dans le cadre SPECIAL CPU REGISTER vous pouvez observer la valeur des registres particuliers du processeur :
  - PC (Program Counter) (ou compteur ordinal) : adresse de la prochaine instruction à exécuter,
  - SP (Stack pointeur) : adresse du pointeur de pile,
  - SR (Status Register) : le registre contenant des informations sur l'exécution de la dernière instruction,
  - BR (Base Register) : le registre de base,
  - SR (Status Flag) : le registre PSW,
  - CPU mode : mode utilisateur ou superviseur,
  - IR (Instruction Register) : le registre d'instruction,
  - MAR (Memory Address Register) : le registre d'adresse associé à la mémoire centrale,
  - MDR (Memory Data Register) : le registre de donnée associé à la mémoire centrale.
- c) Dans le cadre GENERAL PURPOSE CPU REGISTER vous pouvez observer la valeur des registres généraux du processeur.
- d) Dans le cadre PROGRAM LIST vous pouvez observer la liste des programmes chargés en mémoire centrale.
- e) Dans l'onglet Program control vous pouvez choisir la vitesse du processeur (molette slow-fast), puis exécuter les programmes chargés en mémoire centrale avec les boutons RUN (exécution de la totalité du programme) ou STEP (exécution par ligne).

**Exercice 2 — Les états des processus** Le but de cet exercice est de décrire les trois principaux états des processus et d'en identifier les transitions valides.

- 1) Création d'un premier processus :
  - a) Cliquez sur le bouton COMPILER pour ouvrir la fenêtre COMPILER
  - b) Téléchargez le programme BoucleInfini à l'adresse <http://deptinfo.cnam.fr/new/spip.php?rubrique227> et ouvrez le dans l'éditeur de texte du compilateur :

```
program BoucleInfini
  n=0
  while true
    n = n + 1
  wend
end
```
  - c) Compilez ce programme en cliquant sur le bouton COMPILE dans le cadre COMPILER, onglet Compile.
  - d) Enregistrez ce programme en cliquant sur le bouton SAVE du cadre SOURCE sous le nom BoucleInfini.
- 2) Les états des processus et les transitions
  - a) Dans la fenêtre CPU du simulateur, cliquez sur VIRTUAL OS une nouvelle fenêtre s'ouvre :



Sélectionnez l'onglet **VIEWS** et cliquez sur le bouton **VIEW PROCESS STATES**. La fenêtre qui apparaît est une représentation graphique des états des processus. On peut y voir la file des processus qui sont dans l'état prêt (**Ready state**), le CPU où il peut y avoir un seul processus dans l'état élu (**Running state**), et la file des processus qui sont dans l'état bloqué (**Waiting state**).

- b) Dans cette fenêtre, cochez les checkbox **Stay on top** et **Animate**, afin de pouvoir modifier les états d'un processus que vous allez maintenant créer.
- c) Chargez le programme **BoucleInfini** dans la mémoire centrale du simulateur. Pour cela retournez sur la fenêtre du compilateur et cliquez sur le bouton **LOAD IN MEMORY** du cadre **ASSEMBLY CODE**. Vous pouvez observer dans la fenêtre CPU que le programme est chargé dans la mémoire centrale.
- d) Créez maintenant un processus à partir du programme chargé en mémoire. Pour cela retournez dans la fenêtre **VIRTUAL OS**, vous pouvez voir dans la liste **PROGRAM LIST** le programme **BoucleInfini**. Cliquez maintenant sur le bouton **CREATE NEW PROCESS** de l'onglet **Process**. Dans la fenêtre qui représente les états des processus, on peut voir ce processus dans la file des processus prêts avec son numéro (pid).
- e) Dans la fenêtre **VIRTUAL OS** cochez maintenant les trois checkbox **Suspend and state change** dans le cadre **READY PROCESSES**. Cela permettra au simulateur d'arrêter son exécution et d'afficher un message à chaque changement d'état d'un processus.
- f) Pour modifier les états d'un processus, il faut faire des "cliquer-glisser". Si un transition est impossible, le message **\*\*ERROR:Illegal state transition** est affiché en bas de la fenêtre. Cliquez sur le bouton **RESUME**, effectuez les actions suivantes et remplissez les 3 colonnes de droite :

| Actions  | Succès | Échec | État résultant |
|--|--------|-------|----------------|
| Glissez le processus dans la file des processus bloqués ( <b>waiting queue</b> )   |        |       |                |
| Glissez le processus dans la poubelle pour le terminer ( <b>bin</b> )  |        |       |                |
| Glissez le processus dans le CPU. Dans la fenêtre <b>VIRTUAL OS</b> , onglet <b>OS Control</b> , cliquez sur la bouton <b>RESUME</b>   |        |       |                |
| Glissez le processus dans la file des processus bloqués ( <b>waiting queue</b> ). Dans la fenêtre <b>VIRTUAL OS</b> , onglet <b>OS Control</b> , cliquez sur la bouton <b>RESUME</b> |        |       |                |
| Glissez le processus dans le CPU   |        |       |                |
| Glissez le processus dans la poubelle pour le terminer ( <b>bin</b> )  |        |       |                |
| Glissez le processus dans la file des processus prêts ( <b>ready queue</b> ). Dans la fenêtre <b>VIRTUAL OS</b> , onglet <b>OS Control</b> , cliquez sur la bouton <b>RESUME</b>     |        |       |                |
| Glissez le processus dans la file des processus prêts ( <b>ready queue</b> ). Dans la fenêtre <b>VIRTUAL OS</b> , onglet <b>OS Control</b> , cliquez sur la bouton <b>RESUME</b>     |        |       |                |
| Glissez le processus dans la poubelle pour le terminer ( <b>bin</b> )  |        |       |                |

- g) Résumez dans le tableau suivant les transitions d'états possible pour un processus, et dessinez le graphe des états des processus.

| Depuis l'état | Vers l'état |
|---------------|-------------|
|               |             |
|               |             |
|               |             |
|               |             |

### 3) Étude des états des processus

- a) Téléchargez le programme **AttenteLecture** à l'adresse <http://deptinfo.cnam.fr/new/spip.php?rubrique227> et ouvrez le dans l'éditeur de texte du compilateur :

```

program AttenteLecture
  while true
    wait read(caractere)
    writeln(caractere)
  wend
end

```

Chargez le programme en mémoire à l'aide du bouton **LOAD IN MEMORY**.

- b) Dans la fenêtre **CPU** du simulateur, puis dans l'onglet **ADVANCED** cliquez sur le bouton **INPUT OUTPUT**. Une console (ou invite de commande) apparaît, cochez la checkbox **Stay on top**, puis cliquez sur le bouton **SHOW KBD** pour ouvrir une fenêtre clavier.
- c) Dans la fenêtre **VIRTUAL OS**, vérifiez qu'aucune des trois checkboxes **Suspend on state change** ne sont cochées, et que la vitesse du CPU est à la position la plus rapide. Créez un processus avec le bouton **CREATE NEW PROCESS**.
- d) Cliquez sur le bouton **START**, et attendez que le processus soit dans l'état **Waiting** apparaisse dans la console. Dans quel état se trouve le processus ? (on peut le voir dans la fenêtre **CPU**)
- e) En utilisant la fenêtre clavier, cliquez sur n'importe quel bouton. Que se passe-t-il dans le simulateur ?

f) A votre avis que fait ce programme ?

**Exercice 3 — L'ordonnement des processus**      1) Création de processus :

- a) Cliquez sur le bouton **COMPILER** pour ouvrir la fenêtre **COMPILER**
- b) Téléchargez le programme **BoucleOrdo** à l'adresse <http://deptinfo.cnam.fr/new/spip.php?rubrique227> et ouvrez le dans l'éditeur de texte du compilateur :

```
program BoucleOrdo
  i = 0
  for n = 0 to 40
    i = i + 1
  next
end
```

Enregistrez ce programme sous le nom **BoucleOrdo**

- c) Compilez en cliquant sur le bouton **COMPILE** dans le cadre **COMPILER**, onglet **Compile**.
- d) Chargez le programme dans la mémoire centrale du simulateur. Pour cela, cliquez sur le bouton **LOAD IN MEMORY** du cadre **ASSEMBLY CODE**. Vous pouvez observer dans la fenêtre **CPU** que le programme est chargé dans la mémoire centrale.
- e) Créez maintenant un processus à partir du programme chargé en mémoire. Pour cela cliquez sur le simulateur **VIRTUAL OS**. La fenêtre s'ouvre et vous pouvez voir dans la liste **PROGRAM LIST** le programme **BoucleOrdo**. Il est maintenant possible de créer autant de processus que souhaité à partir de ce programme. Pour cela cliquez sur le bouton **CREATE NEW PROCESS**. Créez 3 processus. On peut observer les quatre processus dans la file des processus prêt qui est représentée par la vue **READY PROCESSES**.

2) Politiques d'ordonnement

- a) Sélectionnez la politique d'ordonnement **FCFS** dans **SCHEDULER/Politiques**. Pour activer l'OS, réglez la vitesse au maximum et cliquez sur le bouton **START**. Qu'observez vous quand les processus s'exécutent ? (Regardez spécifiquement **RUNNING PROCESSES** and **READY PROCESSES**). Attendez la fin de l'exécution ou utilisez le bouton **KILL** pour terminer les processus.
- b) Sélectionnez la politique d'ordonnement **Round-Robin (RR)** dans **Scheduler/Politiques**, et l'option **Non-preemptive** dans **Scheduler/Politiques/Priority Type**. Créez 2 processus avec les priorités suivantes : 3 et 2. Pour cela utilisez la liste **Priority** dans **PROGRAM LIST/Processes**. réglez la vitesse à la moitié et cliquez sur le bouton **START**. Pendant l'exécution créez un troisième processus de priorité 1. Qu'observez vous ? Attendez la fin de l'exécution ou utilisez le bouton **KILL** pour terminer les processus.
- c) Sélectionnez maintenant l'option **Pre-emptive** dans **Scheduler/Politiques/Priority Type**. Créez 2 processus avec les priorités 3 et 2. Choisissez 40 **ticks** (40 battements d'horloge) dans **RR Time slice**. Réglez la vitesse à la moitié et cliquez sur le bouton **START**. Pendant que le premier processus s'exécute créez un troisième processus de priorité 1. Qu'observez vous, quelles sont les différences avec la question b) ? Attendez la fin de l'exécution ou utilisez le bouton **KILL** pour terminer les processus.