Simulateur de machine virtuelle - Manuel d'utilisateur

1 Introduction

Ce document est destiné à l'utilisateur final du simulateur. Il s'intéresse à l'utilisation du programme, mais également à différents détails techniques pouvant être utiles à l'utilisateur.

2 Utilisation du simulateur

Pour utiliser le simulateur, il suffit d'ouvrir un terminal et taper la commande

```
$ make all
```

Ensuite, pour compiler et exécuter un programme (demo_sum.txt par exemple), taper :

```
$ ./simulateur demo_sum.txt
```

A noter que la commande lançant le fichier crée un fichier hexa_demo_sum.txt avec le résultat (si elle a réussi) de la compilation.

3 Syntaxe

La syntaxe joue un rôle primordial dans les fichiers assembleur, et certaines choses peuvent vous paraître contre-intuitives si vous n'y êtes pas habitués. Voici une liste non-exhaustive des erreurs de syntaxe non triviales :

3.1 Pas d'espaces entre la définition de l'étiquette et les deux points

Exemple mauvais:

```
debut : in r0
```

Exemple bon:

```
debut: in r0
```

3.2 Pas de saut de ligne après une étiquette

Exemple mauvais:

```
debut:
    in r0
```

Exemple bon:

```
debut: in r0
```

3.3 Les virgules viennent juste après les arguments

Exemples mauvais:

```
add r1 , r0 ,#5
add r1 , r0 , #5
add r1 , r0 ,#5
```

Exemple bon:

```
add r1 , r0 , #5
```

4 Détails techniques

4.1 Valeurs décimales/hexadécimales

Vous pouvez, dans plusieurs contextes, utiliser des valeurs littérales. Pour cela, il vous suffit d'écrire :

#5
#1025
#−2511
#hFFFF

Les valeurs décimales se représentent elle-mêmes (-2511 représente littéralement -2511) cependant les valeurs hexadécimales sont données en tant que représentation complément à 2 de la valeur représentée. Ici, FFFF est la valeur complément à 2 de 1 qui correspond au nombre décimal -1. Ainsi, #hFFFF et #-1 sont équivalents.

4.2 Valeurs maximales autorisées

On dispose de 32 registres (de r0 à r31) donc seulement ceux-ci sont utilisables. Les valeurs littérales sont encodées sur 16 bits, donc puisqu'elles sont signées on peut prendre des valeurs de -32 767 jusqu'à 32 768, celles-ci étant représentées sous leur forme littérale ou hexadécimale en complément à 2.

4.3 Limitations du Cs

Puisqu'on ne peut pas créer de tableau dynamique en C, il nous a fallu faire des choix quant à la taille maximum des chaînes de caractères utilisées tout au long du programme. C'est pourquoi:

- Il y a un maximum de 100 caractères par ligne de code;
- Il y a une taille maximum de 30 caractères pour chaque étiquette;
- Il y a un maximum de 50 étiquettes par programme.

Toutes ces valeurs peuvent être augmentées ou baissées dans le fichier compiler.h en fonction des préférences ou de l'usage de l'utilisateur.

4.4 Noms d'étiquette

Certains mots seront reconnus comme "réservés" par le système, et donc pas utilisables en pratique comme nom d'étiquette

```
      in  r2
      jmp  r1
      out  r1
r1 :  add
```

Ce code ne fait probablement pas ce que vous pensez qu'il fait, puisque l'instruction "jmp r1" est convertie en "jmp 1" et non pas en la bonne adresse. Ainsi, faites attention à ne pas choisir des mots réservés au compilateur en nom d'étiquette.

5 Exemples

Plusieurs exemples ont été mis à votre disposition afin de vous familiariser avec le code assembleur. Vous pouvez également le tester en le compilant et l'exécutant, afin de voir comment fonctionne le programme.