Projet d'Architecture des ordinateurs
Documentation pour l'utilisateur

Tom PIETTON
Anas IBOUBKAREN

Février 2023

1 Avant-propos

Le répertoire `PiettonIboubkaren` est constitué de :

- un fichier `projet.c`
- un répertoire `docs` contenant les fichiers `dev.pdf` et `user.pdf`
- un exécutable `simulateur`

Le programme effectue les tâches suivantes :

① récupérer un fichier texte écrit en code assembleur

② vérifier la syntaxe et la cohérence des données

③ si le fichier en code assembleur ne comporte *aucune erreur*, générer un fichier texte `hexa.txt` comportant la traduction en code numérique de celui-ci

④ exécuter le fichier texte `hexa.txt` s'il a été généré précédemment

Pour compiler un fichier en code assembleur, il faut que vous mettiez le fichier dans le répertoire `PiettonIboubkaren`.

2 Compilation

Vous pouvez trouver un exécutable `simulateur` dans le répertoire `PiettonIboubkaren` qui a été compilé sur une machine du CRIO Unix. En conséquence, nous ne garantissons l'exécutabilité de ce fichier que sur les ordinateurs du CRIO. **Attention : cette version n'est pas la plus récente et détecte mal les erreurs de débordement de pile. Il serait donc préférable de recompiler sur ces machines.**

Pour obtenir un fichier exécutable compatible avec votre machine, nous allons décrire les différentes étapes pour compiler le fichier `projet.c` sur un ordinateur équipé de GNU/Linux sur lequel on peut installer le *GNU Compiler Collection* – GCC.

- Déterminez l'adresse absolue du répertoire `PiettonIboubkaren`.
- Ouvrez le terminal pour actualiser votre répertoire courant grâce à la commande `cd`. Pour cela, faites suivre `cd` d'une **espace** puis de l'**adresse absolue** de `PiettonIboubkaren` et appuyez sur la touche entrée.
- Lorsque votre répertoire courant sera finalement `PiettonIboubkaren`, il vous suffira alors d'écrire sur le terminal `gcc projet.c -o simulateur` puis de taper sur la touche entrée.

Vous obtenez désormais un exécutable du nom de `similateur`.

3 Exécution

Afin d'exécuter ce programme, écrivez sur le terminal `./simulateur` suivi d'une **espace** puis du nom du **fichier en code assembleur** que vous souhaitez compiler et exécuter. Par exemple, si vous voulez compiler le fichier `codeAssembleur.txt`, vous écrirez `./simulateur codeAssembleur.txt` puis appuierez sur la touche entrée.

4 Résultats

Si le fichier en code assembleur contient une ou plusieurs erreurs, des messages précisant leur nature et les lignes où elles se trouvent seront alors affichés. L'affichage se scinde en deux parties. Dans un premier temps, toutes les erreurs relatives à la syntaxe des étiquettes s'affichent avant que le programme ne s'arrête – `hexa.txt` n'est pas généré. Une fois celles-ci corrigées, vous pouvez relancer l'exécution et toutes les erreurs relatives à la syntaxe des mots-clefs et à la cohérence des données sont affichés avant que le programme ne s'arrête – `hexa.txt` n'est pas généré. Lorsque vous n'avez plus aucun message d'erreur, votre code assembleur est jugé correct (**Attention : cela ne signifie pas qu'une erreur ne puisse pas se produire durant l'exécution**) et le fichier texte `hexa.txt` est généré.

Nous exécutons ensuite ce fichier. Le programme s'arrête lors de toute tentative d'accès à une zone mémoire interdite (en dehors de l'intervalle $[0, 4999]$), de toute tentative de division par 0 ou d'accès à une instruction inexistante (dont le numéro est supérieur ou égal au nombre de lignes du fichier `hexa.txt` ou strictement inférieur à 0).