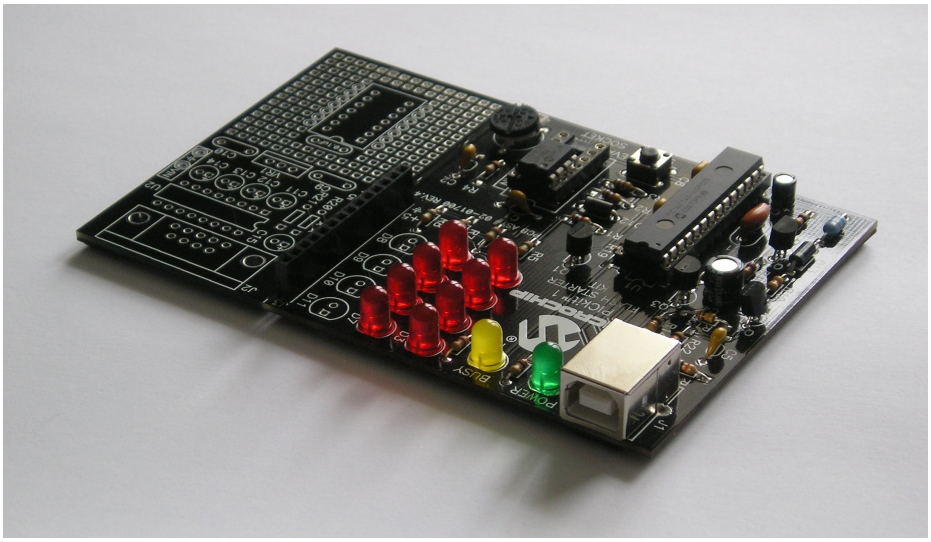
Maxime DICKERSON
Sophie FRANÇOISE
2A SICOM

4 novembre 2013

Projet d'informatique : simulateur MIPS

Livrable n° 2



Professeurs : Nicolas CASTAGNÉ et François PORTET

Introduction

L'objectif de ce livrable est d'ouvrir un fichier ELF. Notre programme doit être capable de charger en mémoire le contenu d'un fichier au format ELF, et d'afficher le contenu de la mémoire. Il doit également pouvoir charger le dictionnaire des instructions, afin de désassembler les instructions et de les afficher.

I – Le dictionnaire d'instructions

Le dossier src/ contient un fichier texte dico.txt qui contient les 25 instructions que notre simulateur doit savoir interpréter.
Nous avons donc écrit la fonction

```
void init_instr(INSTRUCTION* tab);
```

qui parcourt le fichier dico.txt et range chaque instruction dans une structure prévue à cet effet :

```
typedef struct {
        char* nom;      //mnémonique de la fonction
        char* type;     //R I ou J
        int nbe_op;     //nombre d'opérandes
        char* ops[3];   //tableau de taille 3 contenant les opérandes
        unsigned int opcode;
        unsigned int func;
}INSTRUCTION;
```

Nous avons choisi d'utiliser des char* car c'est plus facile à manipuler.
Cette fonction init_inst est exécutée lors du lancement du programme. Afin de vérifier son bon fonctionnement, nous disposons de la commande `inst n` qui permet d'afficher toutes les caractéristiques de l'instruction de la ligne n du dictionnaire. Cette commande n'est donc pas destinée à l'utilisateur du programme. Cela nous permet de vérifier que chaque champ de la structure INSTRUCTION a bien été rempli, avec les bonnes données.

Le dictionnaire, *id est* le tableau DICO, est une variable globale. Cela nous permet d'alléger l'écriture lorsqu'on y fait appel dans une fonction.

De plus, le chargement en mémoire du dictionnaire utilise des pointeurs, donc il faudra vider la mémoire à la fermeture du programme. La fonction

```
void free_memory();
```

est prévue à cet effet, et s'exécute lorsque l'on entre la commande `ex` (fermeture du programme). Elle a été vérifiée lors de son écriture mais à présent on ne peut plus le faire, puisqu'elle s'exécute à la sortie du programme.

II – Chargement d'un fichier ELF

Pour l'ouverture et le traitement d'un fichier ELF, nous avons utilisé ce qui était fourni dans step1. La commande lp <chemin du fichier> permet donc le chargement en mémoire des différentes sections du fichier ELF comme le fait le programme mips-load.

Les tests réalisés ne sont pas encore exhaustifs (il manque certains cas de figure comme par exemple un fichier ELF qui ne soit pas en assembleur MIPS).

III – Affichage de la mémoire

La fonction dm a trois types de syntaxes : soit une seule adresse (on affiche le contenu de l'octet situé à cette adresse), soit un intervalle d'adresses (on affiche les octets qui se trouvent entre les deux adresses), soit une adresse et un nombre entier d'octets à afficher après cette adresse.

Ces trois syntaxes sont reconnues et exécutables dans le cas où tous les paramètres sont de la bonne forme, mais si on demande l'affichage d'un emplacement qui ne contient rien (adresse 0x0 avant qu'un fichier ait été chargé par exemple, ou adresse qui se trouve entre deux sections du fichier ELF), il y a erreur de segmentation. On pourrait envisager d'initialiser les zones mémoires entre deux sections par des 0.

Nous rencontrons également des difficultés pour la réalisation d'un test pour cette fonction dm : c'est le test test/12_dm.simcmd. Il se trouve que le test n'arrive même pas à exécuter la commande lp alors que le test précédent y réussit.

IV – Désassemblage des instructions

Par manque de temps nous n'avons pas commencé à coder la fonction da qui désassemble les instructions.

Conclusion

Dans ce deuxième livrable, nous avons pu finaliser le premier livrable : aucun test n'avait été écrit. Maintenant toutes les fonctions du premier livrable ont des tests qui vérifient leur fonctionnement normal et leur réaction à des mauvais arguments. Cependant, deux des tests que nous avons écrits (09_dr_param qui vérifie le comportement normal de la commande dr et 10_lr fonctionnement normal de la commande lr) se montrent capricieux : ils échouent une fois sur trois ou quatre sans que nous ayons pu en trouver la cause. Néanmoins, nous n'avons jamais constaté ce problème lors de leur exécution « classique » à travers le programme simMips.

En ce qui concerne le deuxième livrable en lui-même, il nous reste encore la commande da à faire et la commande dm à optimiser. Le chargement d'un fichier relogeable ELF et du dictionnaire d'instructions s'effectue sans problème.