Projet : Emulation MIPS[1]

Objectifs : réaliser un émulateur de programmes écrits en langage assembleur MIPS.

[1] Version du 9/11/2015

Description du sujet

Un émulateur est un logiciel capable de reproduire le comportement d'un objet. Dans le cas qui nous intéresse il s'agit de la machine (processeur) MIPS lorsqu'elle exécute un programme. Reproduire le comportement signifie, plus précisément, que l'émulateur gère une mémoire et des registres identiques à ceux de la machine MIPS, puis réalise l'évolution de l'état de cette mémoire et de ses registres au fur et à mesure que les instructions du programme s'exécutent. L'émulateur doit fournir les mêmes résultats que ceux que l'on obtiendrait avec une exécution sur une vraie machine MIPS.

Le processeur MIPS, son architecture et ses instructions sont présentés en annexe.

Deux modes de fonctionnement de l'émulateur sont prévus:

— Le mode dit **interactif** dans lequel chaque instruction MIPS est lue au clavier (entrée par l'utilisateur) et interprétée directement. La commande EXIT (non incluse dans le jeu des instructions MIPS) permet de quitter l'émulateur.

— Le mode dit non-interactif dans lequel l'émulateur va lire les instructions une à une dans le fichier dont le nom (chemin d'accès) est passé en paramètre.

Le projet sera réalisé sous linux. L'émulateur sera appelé en tapant la commande ***emul-mips***:

emul-mips en mode interactif

Une fois cette commande lancée, le programme demande à l'utilisateur d'entrer une instruction, l'exécute et en demande une nouvelle et ainsi de suite jusqu'à lecture de EXIT.

emul-mips prog_filename -pas

où *prog_filename* est le nom d'un fichier contenant le programme écrit en assembleur MIPS à exécuter. Si l'option *–pas* est spécifiée, le programme doit être exécuté pas à pas (c'est-à-dire, demander à l'utilisateur de valider le passage à l'instruction suivante).

Que ce soit en mode interactif ou bien dans le fichier programme, les instructions MIPS sont fournies (une par ligne) en format texte.

L'émulateur commence par afficher l'instruction (ou bien la totalité du programme) sous forme hexadécimale. Par exemple, **2402000a** est la représentation hexadécimale de l'instruction dont le format texte est **addiu $v0, $zero, 10**.

Ensuite, l'émulateur exécutera l'instruction (ou le programme instruction par instruction). Après chaque instruction, l'émulateur va modifier l'état de l'architecture MIPS, c'est à dire les valeurs stockées dans les registres et dans la mémoire. *La précision de votre émulateur est votre priorité*

principale. Plus précisément, assurez-vous que l'état architectural est correctement mis à jour après l'exécution de chaque instruction.

Enfin, l'émulateur affichera le nouvel état (valeur des registres, contenu de la mémoire) avant d'exécuter l'instruction suivante.

Afin de vérifier que votre émulateur fonctionne correctement, vous devez écrire plusieurs programmes à l'aide de toutes les instructions MIPS implémentées.

Travail demandé

Il vous est demandé d'écrire une application C qui réalise un émulateur du processeur MIPS. Cet émulateur reproduira le comportement de chaque instruction[2] et permettra à l'utilisateur d'exécuter des programmes MIPS en observant leurs sorties.

Plus précisément, il vous est demandé d'effectuer dans l'ordre les tâches suivantes::

1. Ecrire des programmes MIPS qui vous serviront à tester votre application. Cette étape est très importante car elle vous permettra de prendre connaissance des spécifications MIPS.
2. Ecrire une première version de l'application permettant de traduire la forme textuelle des instructions MIPS en leur forme hexadécimale et l'afficher.
3. TESTER cette première version aves les tests de l'étape 1.
   Fournitures attendues : les programmes MIPS que vous avez écrits pour tester, l'application que vous avez réalisée (zip du code source et de l'exécutable) et les résultats (sous forme de fichier texte) obtenus par les tests.
4. Définir la structure de votre émulateur en termes de modules; bien préciser comment sont gérés les registres et la mémoire.
   Fourniture attendue : Un document de quelques pages expliquant votre réflexion.
5. Réaliser l'application finale conformément aux décisions de conception que vous avez prises à l'étape précédente.
6. TESTER la version finale.
7. **Fournitures attendues** : l'application que vous avez réalisée (zip du code source et de l'exécutable) et les résultats (sous forme de fichier texte) obtenus par les tests. Un bilan est également demandé précisant **les tâches effectuées par chaque membre de votre groupe**.

[2] Il est possible que toutes les instructions ne soient pas prises en compte, si le temps vous manque. Dans ce cas, il faudra choisir soigneusement des instructions vous permettant, malgré tout, d'écrire des programmes non triviaux.

ANNEXES

1. Document sur le processeur MIPS, extrait du sujet du projet d'informatique de Phelma, 2ème année (François Portet, Nicolas Castagne, Mathieu Chabanas, Laurent Fesquet).
2. Descriptif du jeu d'instruction MIPS.