Mihai-Alexandru DUSMANU

Simulateur de netlists

Structure de données :

Pour retenir les valeurs des variables, j'ai choisi comme structure les tables de hachage (Hashtbl en OCaml) parce que j'avais besoin d'une structure persistante avec des opérations rapides (principalement insertion et recherche, même si j'utilise de temps en temps des suppressions).

Difficultés rencontrées:

I. La partie la plus difficile était de trouver la bonne manière d'implémenter les registres.

Dans mon « scheduler » j'ai choisi de mettre les REG en première. Ainsi, dans le simulateur, au début de chaque itération, je copie les valeurs des registres dans des variables auxiliaires qui vont être utilisées ensuite (à la fin de l'itération) pour mettre à jour les valeurs des registres qui forment des cycles entre eux. Les autres registres dépendent juste des valeurs d'autres variables (qui ne sont pas des registres) donc ils peuvent être mis à jour sans problèmes.

II. Une autre partie difficile était l'implémentation de la mémoire RAM.

Je me suis inspiré de la méthode que j'ai utilisé pour les registres. Dans mon « scheduler » la seule dépendance d'une opération x = RAM<.,.>(readAddr, we, writeAddr, data) est readAddr (l'adresse d'où il faut lire). Dans le simulateur, à la fin de chaque itération je vérifie les valeurs de we, writeAddr et data pour tout bloc RAM et, si we est 1 alors j'écris data dans writeAddr.

Optimisation possible :

Dans la version actuelle du simulateur toutes les variables sont traitées comme des nappes de fils (un fil est une nappe de longueur 1). Cette implémentation a un cout plus grand en mémoire (car on a uniquement des vecteurs, même si c'est un seul fil), mais elle a l'avantage d'être plus pratique – il ne faut pas traiter à coté le cas spécial des opérations entre deux fils.

Informations sur la compilation et les paramètres d'exécution :

Voir le fichier README.pdf / README.md.