Projet Microprocesseur - Partie 1 : simulateur de netlist

November 11, 2021

1 Description du simulateur

1.1 Gestion des variables

Pour stocker les variables pendant toute l'exécution, j'utilise une table de hachage h. Au début je parcours le programme, et j'ajoute une nouvelle variable pour chaque instruction (qui correspond à la variable à gauche du =), et j'initialise chacune de ces variables à 0 (dans les faits à des bits ou paquets de bits $False$). L'initialisation permet d'initialiser les registres (les autres intialisations sont inutiles).

1.2 Gestion des registres

Dans le scheduler, les variables des registres sont ignorées. Les instructions de registre sont ainsi toutes réalisées à la fin. Les variables à mettre dans les registres sont initialisées au début du programme, puis traitées comme toutes les autres variables par la suite.

1.3 Gestion de la RAM

Pour chaque instruction RAM, je crée une nouvelle RAM. L'ensemble des RAM est stocké dans une table de hachage dont les clés sont les identifiants des équations. J'implémente une RAM sous la forme d'un triplet (j'ai créé un type dédié *oneram*) (addr_size, word_size, valeurs), où valeurs est un array des valeurs (formé de sous-array de taille word_size).

Pour la RAM comme pour la ROM, l'accès à des valeurs à l'écriture se fait en convertissant read_addr ou write_addr (qui sont sous forme binaire) en leur écriture en base 10 pour obtenir l'indice dans l'array.

La lecture et l'écriture dans une RAM se fait séparément. Pour éviter des problèmes de lecture et écriture simultanées (on notera que le scheduler ignore les informations sur l'écriture), on réalise toutes les écritures à la fin de chaque étape.

1.4 Gestion de la ROM

Pour la ROM, j'ai préféré ne considérer qu'une seule ROM comme on ne fait que des lectures. Je parcours donc initialement mes équations à la recherche d'une instruction ROM, et j'initialise un array de sous-array de taille word_size en demandant au début de l'exécution les valeurs de la ROM à l'utilisateur mot par mot.

2 Difficultés rencontrées

Apprentissage du Ocaml Comme c'était la première fois que je faisais de la programmation fonctionnelle, j'ai trouvé l'apprentissage du Ocaml au fur et à mesure assez difficile (faire un tri topologique d'un graphe comme premier programme en particulier a été laborieux).

Gestion de la RAM (et de la ROM) La gestion de la RAM me paraissait assez floue au début. J'ai recontré plusieurs problèmes dans l'implémentation, notamment pour la gestion séparée de la lecture et de l'écriture, et plus généralement pour le choix de la structure de donnée.

Gestion des registres J'ai eu du mal à voir comment traiter les registres dans le scheduler.