Rapport du Projet de Système Digital n°2

Ismaël Belghiti, Damien Desfontaines, Guillaume Geoffroy, Kenji Maillard

1 Introduction

Nous avons fini la création du simulateur de circuit et du langage de description du circuit. Depuis le rapport n°1, le langage n'a évolué que pour ajouter quelques fonctionnalités supplémentaires d'ordre pratique, comme le fait de coder directement une certaine suite binaire avec une syntaxe du type \$010011, la possibilité de commenter le code source ou de le séparer en plusieurs fichiers. Nous ne nous attarderons pas sur ces modifications mineures.

Ce rapport décrit donc l'étape suivante, c'est à dire le dessin du microprocesseur et ses caractéristiques techniques.

2 Inspiration et dessin du microprocesseur

Pour le dessin du microprocesseur, nous nous sommes inspirés des schémas de fonctionnement MIPS présents dans l'ouvrage Computer Organization and Design. Nous donc reprenons ici le schéma global de fonctionnement, tiré de cet ouvrage :

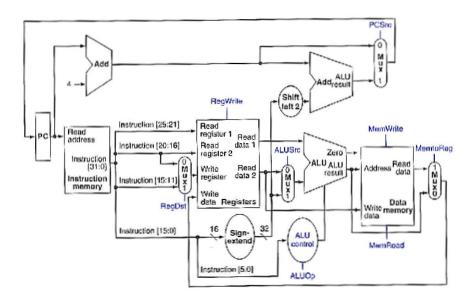


Fig. 1:

Décrivons les différents blocs présents sur ce schéma qu'il a fallu implémenter.

2.1 ALU

Principe : L'ALU, Unité Arithmétique et Logique en français, est la partie du microprocesseur chargée d'effectuer les calculs. Elle reçoit un identifiant d'opération du bloc "ALU Control", effectue l'opération correspondante sur ses deux entrées et renvoie le résultat. Il s'agit donc d'un bloc purement combinatoire : rien n'est stocké dans l'ALU, qui ne contient donc aucun registre et effectue toutes les opérations en un seul cycle.

Opérations implémentées : Les opérations implémentées sont les opérations arithmétiques et logiques classiques : + , - , * , And, Or, Xor, Nand, Nor (la division n'est pour l'heure pas incluse). Le fichier Alu.rock présente ce bloc, les commentaires à la fin du fichier devraient être suffisants pour pouvoir lancer quelques tests.

2.2 Registres

Le bloc de registres contient 32 registres, dont l'organisation et le fonctionnement sont calqués sur le modèle MIPS. Ou pas. TODO

2.3 Control

Le bloc de contrôle, présent dans notre projet sous le nom de "Décodeur" a dans notre projet non seulement le rôle du bloc "Control" de l'architecture standart MIPS mais c'est aussi lui qui s'occupe de la synchronisation entre les différents cycles parcourus pour une instruction donnée. TODO à détailler

2.4 Instruction Memory

TODO

2.5 Data Memory

La mémoire va être implémentée par un device. TODO à détailler

3 Détails techniques

3.1 Format des instructions

TODO explication du R-format, I-format, format des jumps, etc.

3.2 ALU

 $\textbf{Fonctionnement}: \ \ Le \ bloc \ Alu{<}m{>} \ du \ code \ source \ prend \ 3 \ entrées:$

- deux arguments a et b sur m bits,
- un sélecteur d'opération sur 4 bits,

et en fonction de la valeur du sélecteur, qui contient le "code" de l'opération voulue (0 pour And, 1 pour Or, etc.), il renvoie le résultat sur m bits, tronqué si besoin (les éventuelles retenues sont perdues, tous les calculs sont faits modulo 2^m).

Comment ça marche: Chaque opération de l'ALU est codée séparément. Puis, à la lecture de deux entiers, toutes les opérations possibles sont exécutées simultanément, et un filtre sélectionne le résultat voulu. Tout fonctionne en un seul cycle, et les circuits correspondant aux diverses opérations ont tous été codés récursivement.

3.3 Registres

TODO, etc.