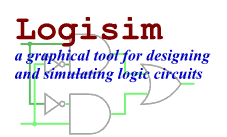
**Benjamin Dutriaux**

**Rapport projet Logisim**



*Licence 3 MIAGE groupe 509H*

**I - Implémentation du Nono-1**

**Les différents circuits implantés dans logisim :**

**UAL**

C’est l’unité d’arithmétique logique, c’est très simple, elle reçoit en entrée les deux valeurs de registre, et le une entrée sélectionnant l’opération à effectuer.

Elle peut donc ajouter, soustraire, faire une ET logique, un OU logique, un NOT, un décalage à droite ou un décalage à gauche entre les deux valeurs de registre.

L’entrée qui décide de l’opération est reliée à un multiplexeur qui sélectionne alors l’opération à effectuer.

Elle possède aussi une sortie correspondant aux flags, il s’agit de l’overflow flag, indiquant que le nombre est trop grand pour être affiché, le zero flag, indiquant que le résultat est égal à 0, le sign flag indiquant le signe du résultat (0 pour positif, 1 pour négatif) et le carry flag indiquant si il y a un bit supplémentaire devant le résultat.

**UAL\_OF**

Il s’agit du circuit gérant l’overflow flag de l’UAL.

**Decodeur\_instruction**

Il s’agit du circuit qui décode les instructions, il prend en entrée l’opcode, et selon cela, il détermine le code à envoyer à l’UAL (sortie ctrlUAL), il détermine aussi si l’instruction est un saut (isJMP), si les registres doivent autoriser l’écriture (regWrite), et si l’instruction est un chargement (isLoad).

**RegWrite\_Decodeur**

C’est le circuit déterminant si les registres doivent être en mode écriture pour cette instruction.

Cette partie est séparée car elle était beaucoup plus complexe à la base mais a été simplifiée.

**isLoad\_Decodeur**

C’est le circuit déterminant si l’opération est un chargement, si c’est un chargement, le banc de registre prend sa valeur de la RAM, sinon il la prend de l’UAL.

Cette partie est séparée car elle était beaucoup plus complexe à la base mais a été simplifiée.

**isJmpDecodeur**

C’est le circuit déterminant si l’opération est un saut, déterminant ainsi la valeur que le sélecteur de registre doit utiliser.

Cette partie est séparée car elle était beaucoup plus complexe à la base mais a été simplifiée.

**ctrUAL\_Decodeur**

C’est le circuit déterminant le code à rentrer dans l’UAL qui l’utilisera pour choisir l’opération à effectuer.

Cette partie est séparée car elle était beaucoup plus complexe à la base mais a été simplifiée.

**contrôleDeSaut**

Le contrôle de saut détermine si l’opération doit être un saut, doit éteindre le système ou si l’on doit passer à l’instruction suivante. Elle prend en entrée l’opCode et les flags, selon l’opcode, elle détermine l’action à effectuer, et si c’est un saut elle doit regarder les flags et en fonction de ceux-ci déterminer la nature du saut.

**bancDeRegistre**

Le banc de registre prend en entrée la valeur de l’instruction calculée par l’UAL, trois sélecteurs de registre indiquant dans quel registre écrire et depuis lesquels il doit charger une valeur. Les sélecteurs sont reliés à un multiplexeur pour effectuer le choix des registres.

Les registres sont reliés à un reset qui peut les remettre à 0 et a une horloge qui les cadence.

**selecteurDeRegistre**

Le sélecteur de registre reçoit en entrée des valeurs qu’il doit interpréter en fonction de l’indicateur isJmp, ensuite, il envoie ces valeurs au banc de registre, ce qui détermine les registres en écriture, et en lecture.

**Main**

La partie main intègre tous ces sous-circuits, et ajoute des éléments, la RAM, les sauts, et des multiplexeurs qui sélectionnent les entrées et sorties devant être pris en compte.

**II - Utilisation de Nono-1**

1.

Pour implémenter l’instruction halt, on utilise un multiplexeur dont une entrée est branchée sur une constante FF, quand l’entrée sélectionnée est celle-ci, le registre PC prend cette valeur, ensuite le système reconnait cette valeur et va donc exécuter l’instruction halt qui aura pour but d’éteindre le système.

Il faut faire attention à ne pas utiliser cette valeur dans nos programmes car celle-ci mettrais le système à l’arrêt.

**Liste des tâches inachevées :**

Je n’ai pas utilisé les flags dans le contrôleur de saut car je n’ai pas réussi à comprendre la logique.

Je n’ai pas non plus réalisé la fin de la partie 2 par manque de temps.

**Les limitations du programme :**

Chaque partie semble fonctionner correctement indépendamment (hormis le contrôle de saut, comme expliqué plus haut), mais je n’ai pas eu le temps de tester le circuit dans son ensemble.

**Les difficultés rencontrées :**

Le contrôle de saut dont je n’ai pas su comprendre la logique.

La découverte de certains éléments de logisim un peu tard qui auraient pu simplifier certains circuits (Tunnel …).