Projet d'architecture et circuits

1 Présentation

Le but de ce projet est de construire avec `logisim` un processeur qui implante certaines instructions du LC-3. Le point de départ est le circuit `logisim` fourni nommé `LC-3.circ`, que vous pouvez trouver sur l'espace Didel du cours [1].

1.1 Instructions à implanter

Ce circuit implémente déjà les instructions `NOT, ADD, AND` du LC-3. Il reste à ajouter des composants au circuit pour qu'il implémente d'autres instructions :

- Minimum requis : "câbler" les instructions `LEA, LDR, STR, LD, ST, BR, JMP`, ainsi que le `ADC` décrit ci-dessous.
- Projet avancé : ajouter les instructions `JSR` et `JSRR`.
- Projet très avancé : coder `LDI` et `STI`.

1.2 L'instruction ADC

Le but est d'étendre le LC-3 avec une nouvelle instruction `ADC` (pour "ADD with Carry"), permettant de faire une addition avec une retenue éventuelle, et d'implémenter cette instruction en `logisim`.

Motivation. Cette instruction `ADC` est utile si on souhaite simuler des nombres 32-bits (ou plus) à l'aide de nombres 16-bits : Chaque nombre 32-bits sera représenté en deux blocs de 16-bits (mis par exemple dans deux registres du LC-3), et une addition se fera alors bloc par bloc, avec une propagation de retenues entre les blocs.

Description. Un `ADC` se comportera comme un `ADD`, à ceci près qu'une retenue de 1 sera ajoutée lorsque la précédente opération arithmétique aura généré un débordement. En pratique, seules les instructions ADD et ADC peuvent entraîner des débordements, et donc des additions avec retenues par la suite.

État d'overflow. Les débordements seront stockés sous forme d'un état `o` (pour "overflow"), de fonctionnement similaire aux états `nzp`, mais qu'une instruction BR ne peut pas tester directement. L'état `o` devra :

- être mis à 1 après un `ADD` ou `ADC` ayant entraîné un débordement
- être mis à 0 après tout autre `ADD` ou `ADC`, ou bien après une autre instruction modifiant `nzp`
- être préservé par toute autre instruction.

Codage de l'instruction ADC. L'opcode de `ADC` sera `1101`. L'instruction `ADC` doit modifier les états `nzp`, ainsi que l'état `o`. Elle supportera deux modes d'adressage calqués sur ceux de `ADD` :

15			12	11 9	8 6	5	4 3	2 0
1	1	0	1	DR	SR1	0	0 0	SR2
1	1	0	1	DR	SR1	1	Imm5	

La première ligne correspond à l'adressage registre : `DR` reçoit le résultat de l'addition entre `SR1` et `SR2`, plus 1 si l'état `o` vaut 1. La deuxième ligne correspond à l'adressage immédiat : `DR` reçoit le résultat de l'addition entre entre `SR1` et `Imm5`, plus 1 si l'état `o` vaut 1.

1.3 Les documents de référence

Les documents de référence pour la description du LC-3 sont :
- le cours [2], en particulier "Cours n° 6 : description du LC-3".
- les TD7 et ultérieurs, ainsi que le document `ISA.pdf` disponibles sur l'espace Didel du cours [1]. Vous trouverez bientôt au même endroit des programmes d'exemples.
- le site internet du livre de Patt & Patel [3], voir en particulier l'annexe A.

2 Paquetage

Le projet à rendre doit contenir :

2.1 Fichiers

Le fichier `logisim` implémentant le processeur LC-3, ainsi que vos programmes de test (de type `.mem`). Vous pouvez modifier à votre guise le fichier fourni, que ce soit par ajout ou réorganisation. Conservez simplement les noms des labels existants et la partie "Traces" du circuit principal, qui sera utilisée pour des tests automatiques. Les composants fournis mais incomplets sont signalés par une bordure rouge. N'hésitez pas à ajouter des commentaires explicatifs dans le fichier.

2.2 Mémoire

Vous devez rendre un mémoire expliquant la démarche suivie. Il faut énumérer rapidement les modifications et composants ajoutés au circuit et montrer qu'ils réalisent bien les fonctions souhaitées. Votre mémoire doit aussi comporter le code de vos programmes de test (désassemblés) et justifier qu'ils illustrent le fonctionnement du simulateur et l'intérêt des nouvelles instructions. Le mémoire ne devra pas dépasser 5 pages.

3 Modalités pratiques

Le projet s'effectue seul ou en binôme. Il donnera lieu à une soutenance *individuelle* en janvier. La note tiendra compte de la lisibilité du circuit et de la simplicité de la solution choisie.

3.1 Remise du projet

Le rendu du projet se fera via l'espace Didel du cours (ou bien en cas de problème par mail : `pierre.letouzey@inria.fr`). Le projet devra être rendu vers la fin de la session d'examen de janvier, la date limite précise vous sera précisée ultérieurement.

Références

[1] `http://didel.script.univ-paris-diderot.fr/claroline/course/index.php?cid=M1ARCHI`

[2] Oliver Carton : `http://www.liafa.jussieu.fr/~carton/Enseignement/Architecture`

[3] Yale N. Patt, Sanjay J. Patel, Introduction to Computing Systems : From Bits and Gates to C and Beyond, McGraw-Hill, `http://highered.mcgraw-hill.com/sites/0072467509/`