Les composantes du processeur multi-cycles

Ce document doit être lu en visualisant le circuit le plus complet du processeur multi-cycles présenté en classe afin de bien comprendre les explications comprises dans ce document.

Si une composante du processeur multi-cycles n'est pas décrite dans ce document, veuillez vous référer au document sur le processeur à 1 cycle. Cela signifie que la composante fonctionne de la même façon.

Les composantes principales

Le module d'accès à la mémoire (*Memory*)

Ce module accède à la mémoire pour aller chercher une valeur à l'adresse donnée par l'entrée *Address*. Contrairement au processeur à 1 cycle, ce module sert autant à la lecture des instructions qu'à la lecture ou l'écriture des données. La donnée à écrire, dans le cas d'une écriture, est fournie à travers l'entrée *Write data*. La valeur lue en mémoire est ensuite écrite dans le registre *Memory data register* et dans le registre *Instruction register* (si le bit de contrôle de l'*Instruction register* permet l'écriture).

Les registres

Instruction register

L'instruction est lue en mémoire durant le premier cycle de l'exécution d'une instruction. À la fin du premier cycle, l'instruction est écrite dans le registre *Instruction register* et son contenu sera accessible durant toute l'exécution de l'instruction.

Memory data register

Dans le cas d'une instruction de lecture en mémoire, la lecture de la mémoire est faite au 4ème cycle de l'instruction. Au 5ème cycle, nous allons sauvegarder la valeur lue dans un registre du banc de registres. Le registre *Memory data register* sert à conserver la valeur lue en mémoire du 4ème cycle au 5ème cycle de l'instruction. Ce registre est affecté par chaque lecture en mémoire même si sa valeur n'est significative pour le circuit que lors d'une instruction de type chargement.

A

Ce registre contient la première valeur lue dans le banc de registres pour cette instruction afin d'être utilisée au cycle suivant.

B

Ce registre contient la deuxième valeur lue dans le banc de registres pour cette instruction afin d'être utilisée au cycle suivant.

ALUOut

Ce registre contient le résultat du calcul fait par l'unité arithmétique et logique afin d'être utilisé au cycle suivant.

Les bits de contrôle

PCWriteCond

Ce bit est activé lorsqu'on exécute une instruction de branchement conditionnel (*beq*). Il est équivalent au bit *Branch* du processeur à 1 cycle.

PCWrite

Comme nous exécutons une instruction avec plusieurs cycles (3 à 5), nous n'avons pas besoin d'écrire une valeur dans le registre PC à chaque cycle. Ce bit sert à indiquer quand il est permis d'écraser la valeur actuelle du registre PC avec une nouvelle valeur (PC + 4 par exemple).

IorD

Ce bit sert à déterminer la source de l'adresse à laquelle nous désirons accéder en mémoire. Plus concrètement, ce bit indique s'il s'agit de l'adresse d'une instruction ou de l'adresse d'une donnée qui est reçue par le module d'accès à la mémoire.

IRWrite

La valeur du registre *Instruction register* ne doit être écrasée que lors du premier cycle de chaque instruction. Pour éviter d'écraser la valeur du registre à chaque accès à la mémoire, nous avons ce bit qui indique au registre s'il doit prendre une nouvelle valeur ou non. Le registre *Instruction register* doit contenir l'instruction jusqu'au dernier cycle de l'exécution de cette instruction.

ALUSrcA

Ce bit sert à déterminer la première opérande envoyée à l'UAL. Il peut s'agir de la valeur du registre A ou bien de la valeur du registre PC. La valeur du registre PC est envoyée à l'UAL lorsqu'on veut calculer l'adresse de la prochaine instruction ou lorsqu'on veut calculer le *Branch Target*.

ALUSrcB

Ce signal sur 2 bits sert à déterminer la deuxième opérande envoyée à l'UAL. Il peut s'agir de la valeur du registre B. Il peut s'agir de la valeur 4 dans le cas du calcul de l'adresse de la prochaine instruction (PC + 4). Il peut s'agir d'une valeur immédiate sur 32 bits ou bien d'une valeur immédiate sur 32 bits ayant subit un *shift* à gauche de 2 bits (pour calculer le *Branch Target*).