1. Dans le circuit mult\_2, les offsets sont incrémentés de 1 au lieu d’être incrémenté de 2 dans le

circuit non-pipeliné, pourquoi?

2. Dans le circuit fetch, le signal LR\_adr\_o vient d’un registre et est connecté au bloc decode au

lieu du bloc bank\_register, pourquoi ?

Cela permet de ne pas avoir à attendre la fin du WB de l’instruction précédente pour accéder à la valeur du LR

3. Dans le circuit decode, le signal adr\_reg\_d\_s est mis dans un registre alors que les signaux

adr\_reg\_n\_s, adr\_reg\_m\_s et adr\_reg\_mem\_s sont directement connecté à la sortie, pourquoi?

On regarde si adr\_reg\_d\_s contient une adresse. Si oui, cela veut dire qu’on écrit dedans et que qu’il faut attendre qu’il contienne sa nouvelle valeur avant de continuer le pipeline.

On écrit rien dans les autres registres donc pas besoin de stocker leur adresse

4. Dans le circuit decode, les signaux des bus de contrôle sont connectés aux registres avec une

porte MUX contrairement aux autres signaux, pourquoi ?

Les bus de contrôles sont chargés de dire aux différents blocs ce qu’ils doivent faire. Or certaines de nos instructions peuvent provoquer des aléas, donc nous devons pouvoir figer la pipeline le temps que l’aléa se résolve.

Par exemple, on dispose d’un enable pour notre DECODE qui se chargera de figer la suite du processeur

5. Si on voulait ajouter le multiplieur 5x3 pipeliné du laboratoire préparatoire, quelles seraient les

conséquences sur le pipeline du processeur ? Comment ça pourrait être fait ?

Si le but est que les instructions de multiplication soient effectuées par ce multiplier, et bien cela ralentirait notre processeur. Car chaque instruction MUL devrait se faire sur 3 cycles en plus des cycles de pipeline de notre processeur.