Corrigé Série TD n°5.2

Exercice n°1 :

1.1 Pipelining égal

$$Temps\ de\ cycle_{Pipeliné} = \frac{Temps\ de\ cycle_{Non\ pipeliné}}{Nombre\ d'étages\ du\ pipeline} + Temps\ de\ stabilisation$$

1) Temps $cycle_{pipeliné}$ = temps cycle $_{non\ pipeliné}$ / Nombre d'étages pipeline + surcoût
   En appliquant cette formule, on obtient : 5.5ns, 3ns, 1.75ns et 1.125ns respectivement.
2) Latence de chaque processeur = temps de cycle * nombre d'étages
   ce qui donne 11ns, 12ns, 14ns et 18ns respectivement.
3) Nombre d'étages pipeline = temps cyclenonpipeliné / Temps cyclepipeliné – surcoût
   L'application de la formule nous donne 6.67 étages ≈ 7 étages

1.2 Pipelining inégal

– Quel est le temps de cycle minimal qui peut être atteint en implémentant le pipeline sur ce processeur ?

Correction : *Le temps de cycle minimal est déterminé par le temps d'exécution de l'étage le plus lent, auquel il convient d'ajouter le temps de stabilisation. Ici, l'étage 4 est le plus lent (7 ns), le temps de cycle est donc :* 7+1 = 8 *ns.*

– Si le processeur est divisé en un plus petit nombre d'étages qui lui permettent toutefois d'atteindre le temps de cycle de la question précédente, quel sera le temps d'exécution d'une instruction complète ?

Correction : *On peut regrouper dans un unique étage n'importe quel ensemble de modules adjacents qui offriraient un temps d'exécution total inférieur ou égal à 7 ns. Ceci nous permet de créer un pipeline à 5 étages. Le temps d'exécution d'une instruction complète est alors* 8×5 = 40 *ns. (Cycle d'horloge* × *nombre d'étages)*

– Si le pipeline est limité à deux étages, quel est le temps de cycle minimal ?

Correction : *Pour pouvoir obtenir un temps de cycle minimal, il faut diviser les modules en étages ayant des temps d'exécution aussi égaux que possible. Pour deux étages, on obtient des temps de 16 et 9 ns (ou l'inverse). Le temps de cycle minimal est alors de 17 ns. (*16 *ns*+1 *ns de stabilisation).*

– Quel est alors le temps d'exécution d'une instruction complète ?

Correction : *Le temps d'exécution d'une instruction complète est alors* 2×17 = 34 *ns.*

Exercice n°2 :

DIV r2, r5, r8
SUB r9, r2, r7
ASH r5, r14, r6
MUL r11, r9, r5
BEQ r10, #0, r12
OR r8, r15, r2

– Identifiez tous les aléas LAE (Lecture Après Écriture).
Correction : *Aléas LAE entre DIV et SUB (r2), AND et MUL (r5), SUB et MUL (r9) et DIV et OR (r2).*
– Identifiez tous les aléas EAL (Écriture Après Lecture).
Correction : *Aléas EAL entre DIV et AND (r5) et DIV et OR (r8).*
– Identifiez tous les aléas EAE (Écriture Après Écriture).
Correction : *Pas d'aléa EAE.*
– Identifiez tous les aléas de contrôle.
Correction : *Il n'y a qu'un aléa de contrôle entre BEQ et OR.*

Exercice n°3 :

Il n'existe aucun aléa dans ce code, aussi, les instructions traversent le pipeline à la cadence d'un étage par cycle

	Cycles							
	1	2	3	4	5	6	7	8
EI	ADD	SUB	MUL	DIV				
DI		ADD	SUB	MUL	DIV			
LR			ADD	SUB	MUL	DIV		
EX				ADD	SUB	MUL	DIV	
ER					ADD	SUB	MUL	DIV

Cette fois, le programme ***comporte un aléa LAE, entre l'instruction SUB, qui écrit r4 et l'instruction MUL qui le lit****. L'instruction MUL ne sera donc pas capable de lire ses registres d'entrée avant que l'instruction SUB n'ait terminé l'étage ER, ce qui crée un blocage de pipeline.*

	Cycles									
	1	2	3	4	5	6	7	8	9	10
EI	ADD	SUB	MUL	DIV						
DI		ADD	SUB	MUL	DIV	DIV	DIV			
LR			ADD	SUB	MUL	MUL	MUL	DIV		
EX				ADD	SUB	nop	nop	MUL	DIV	
ER					ADD	SUB	nop	nop	MUL	DIV