Pipeline aléas

ARO
8 - Pipeline

Résumé du document

TBD

Table des matières

1. Dépendances

Deux instructions ont une dépendance si elles ont besoin de la même chose pour fonctionner correctement, comme un même registre, une même unité de calcul ou une même adresse mémoire. Lorsque cela se produit, le pipeline doit attendre que l'instruction précédente termine de travailler sur cette ressource avant de pouvoir commencer à travailler sur l'instruction suivante. Cela peut entraîner des retards et réduire l'efficacité du pipeline.

Les aléas sont inhérents au parallélisme des instructions et peuvent être classés en trois catégories principales :

1.1. Dépendances structurels

Ce type de problèmes survient lorsque deux instructions dans des étages différents du pipeline nécessitent la même ressource.

1.1.1. Exemple

Dans l'exemple ci-dessous nous voyons que l'instruction LDR nécessite l'accès à la mémoire pour charger une valeur dans un registre, tandis que l'instruction ADD nécessite l'accès à la mémoire pour stocker le résultat d'une addition. Si ces deux instructions sont exécutées en même temps, elles entreront en conflit pour accéder à la mémoire, ce qui entraînera un aléa structurel.

	1	2	3	4	5	6	7	8
LDR R7,R6,0	IF	ID	EX	MA	WB			
ADD R6,R6,1		IF	ID	EX	MA	WB		
ADD R0,R0,1			IF	ID	EX	MA	WB	
ADD R1,R1,1				IF	ID	EX	MA	WB

	1	2	3	4	5	6	7	8	9
LDR R7,R6,0	IF	ID	EX	MA	WB				
ADD R6,R6,1		IF	ID	EX	MA	WB			
ADD R0,R0,1			IF	ID	EX	MA	WB		
ADD R1,R1,1					IF	ID	EX	MA	WB

1.2. Dépendances de données

Accès simultané aux mêmes registres.

1.2.1. RAW (Read After Write)

L'instruction n+x lit une source modifiée par l'instruction n.

```
ADD R1, R2, R3          R1 = R2 + R3
SUB R5, R1, #2          R5 = R1 - 2
```

1.2.2. WAR (Write After Read)

L'instruction n+x écrit dans une destination que l'instruction n utilise comme source.

```
ADD R1, R2, R3          R1 = R2 + R3
SUB R2, R4, #2          R2 = R4 – 2
```

1.2.3. WAW (Write After Write)

L'instruction n+x écrit dans une destination et l'instruction n écrit dans cette même destination.

```
ADD R1, R2, R3          R1 = R2 + R3
AND R5, R1, R2
SUB R1, R4, #2          R1 = R4 – 2
```

1.2.4. Exemple

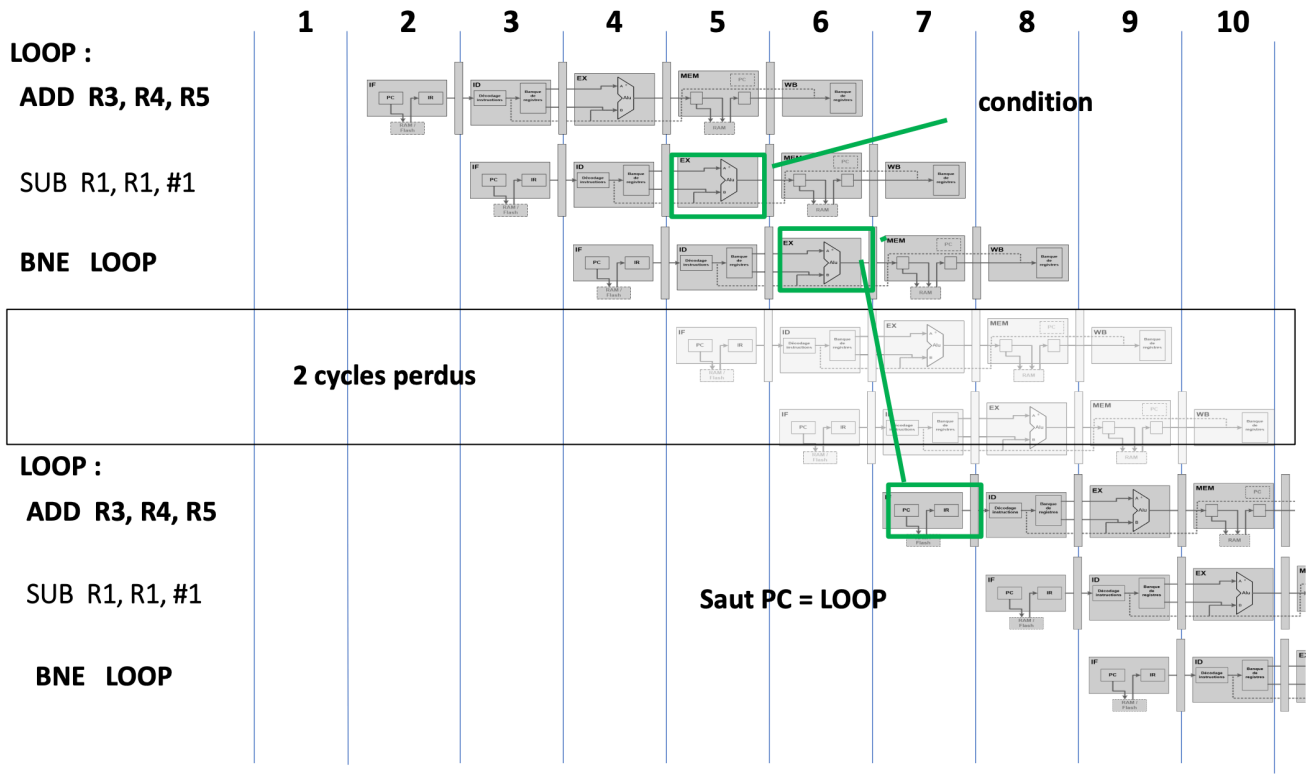
Ce type de problèmes survient lorsque deux instructions dans des étages différents du pipeline nécessitent le même registre alors que ce dernier n'a pas encore été mis à jour.

	1	2	3	4	5	6	7	8
ADD R1,R1,1	IF	ID	EX	MA	WB			
ADD R0,R1,R2		IF	ID	EX	MA	WB		

Vu qu'il faut attendre l'opération WB (Write Back) pour que dans le registre `R1` la valeur de l'addition `R1` et 1 soit stockée, la deuxième instruction ADD ne peut pas être exécutée avant que la première instruction ADD ne soit complètement terminée.

1.3. Dépendances de contrôle

Lors de l'exécution d'une instruction de branchement conditionnel, on dit que le branchement est pris si la condition est vérifiée et que le programme se poursuit effectivement à la nouvelle adresse.



2. Arrêt de pipeline

Pour éviter les aléas, le pipeline peut être arrêté. Cela peut se faire de différentes manières :

2.1. Arrêt de pipeline par hardware

Lorsqu'un aléa est détecté, le pipeline peut être arrêté pour attendre que l'aléa soit résolu. Cela peut se faire en vidant le pipeline ou en insérant des bulles.

	1	2	3	4	5	6	7	8	9	10
ADD R1, R2, R3	IF	ID	EX	MEM	WB					
SUB R5, R1, #2		IF	ID	ID	ID	ID	EX	MEM	WB	

2.2. Arrêt de pipeline par software

	1	2	3	4	5	6	7	8	9	10
ADD R1, R2, R3	IF	ID	EX	MEM	WB					
NOP		IF	ID	EX	MEM	WB				
NOP			IF	ID	EX	MEM	WB			
NOP				IF	ID	EX	MEM	WB		
SUB R5, R1, #2					IF	ID	EX	MEM	WB	