Universidade Federal de Roraima
Departamento de Ciência da Computação
Arquitetura e Organização de Computadores

DISCIPLINA: ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES
ALUNO: Talles Bezerra de Assunção

Lista de Exercício 02

1) Quais as vantagens de um processador multiciclo em relação a um uniciclo?

Como no multiciclo as instruções são divididas em passos e cada passo gasta um ciclo de clock, não a necessidade do ciclo de clock ter a mesma duração para todas a instruções, pois cada instrução usa apenas o número de ciclos que ela necessita.

2) Quais as modificações necessárias em um processador multiciclo simples para que se introduza a função de pipeline?

Cada instrução no multiciclo é executada em 5 estágios: busca de Instrução na Memória (IF), leitura de registradores e decodificação da instrução (simultâneos) (ID), execução da operação ou cálculo de endereço (EX), acesso de um operando na memória (MEM), escrita de resultado em um registrador (WB), sem pipeline a próxima instrução só começa a ser executada quando a anterior tiver passado por todos os 5 estágios, com pipeline, assim que um estágio de uma instrução termina ele passa a instrução para o próximo estágio e inicia a execução da próxima instrução com o estágio que não vai mais ser utilizado pela instrução anterior, trabalhando mais de uma instrução no mesmo ciclo.

Exemplo:

Inst1	IF	ID	EX	MEM	WB	
Inst2		IF	ID	EX	MEM	WB
Inst3			IF	ID	EX	MEM	WB

3) Considerando o pipeline do MIPS (simples com MEM compartilhada para instrução e dados) e uma iteração de loop conforme o trecho de programa abaixo, relacione os conflitos que podem ocorrer e seus consequentes stalls. Qual o speedup (por iteração) para o programa em relação à versão sem pipeline?

```
Loop: subi $t2, $t2, 4
      lw $t1, 0($t2) – Conflito de dados, necessita do $t2
      add $t3, $t1, $t4 – Conflito de dados, necessita do $t1
      add $t4, $t3, $t3 – Conflito de dados, necessita do $t3
      sw $t4, 0($t2) – Conflito de dados, necessita do $t4
      beq $t2, $0, loop
```

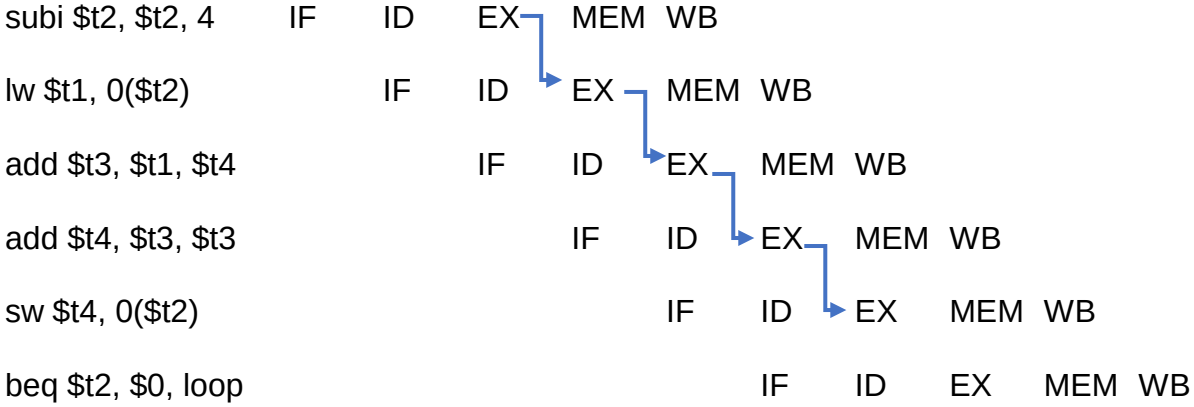
Ciclo de clock: 1ns
Sem pipeline:
3 ciclos para instruções BEQ, 4 ciclos para instruções de tipo R e SW, e 5 ciclos LW.
Subi 4 ciclos
Lw 5 ciclos
Add 4 ciclos
Add 4 ciclos
Sw 4 ciclos
Beq 3 ciclos
Total 24 ns

Com pipeline:

1C	2C	3C	4C	5C	6C	7C	8C	9C	10C	11C	12C	13C	14C	15C	16C	17C	18C	19C	20C	21C	22C
IF	ID	EX	MEM	WB																	
	IF				ID	EX	MEM	WB													
		IF							ID	EX	MEM	WB									
			IF										ID	EX	MEM	WB					
				IF													ID	EX	MEM	WB	
					IF													ID	EX	MEM	WB

Total 22 ns
Speedup = 24/22 = 1.09 ns

4) Na questão anterior, assume que a memória de instruções e dados podem ser segmentadas e que o processador aplica a técnica de bypassing. Como ficarão os conflitos e seus stalls, e o speedup?



Speedup = 5 / (1 + 66,7% x 4)
Speedup = 5 / 3.668 = 1.36

5) No programa abaixo, relacione as dependências (dados, WAR, WAW e outros) existentes.

```
div.d   F1, F2, F3
sub.d   F4, F5, F1
s.d     F4, 4(F10)
add.d   F5, F6, F7
div.d   F4, F5, F6
```

Dependência RAW

- Div.d e sub.d – Uso do F1
- Sub.d e s.d – Uso do F4
- Add.d e div.d – Uso do F5

Dependência WAR

- Add.d pode escrever o registrador F5 que sub.d lê
- Div.d pode escrever o registrador F4 que s.d lê

Dependência WAW

- Sub.d pode terminar depois de div.d – Uso do F6