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

LISTA DE EXERCÍCIO 02

ATENÇÃO: Descrever as soluções com o máximo de detalhes possível. Todos os artefatos (relatório, código fonte de programas, e outros) gerados para este trabalho devem ser adicionados em um repositório (com o seguinte formato nome_ufrr_AOC_2018_2) no site github.com.

PRAZO DE ENTREGA: 06/12/2018

1) Quais as vantagens de um processador multiciclo em relação a um uniciclo?
R - Faz com que as instruções sejam executadas em quantidade de períodos de clocks diferentes, e pode compartilhar unidades funcionais, pelo fato de as fases de execução de uma instrução ser separadas e executadas de forma independente, dando aproveitamento de Hardware, dando suporte assim à pipeline.

2) Quais as modificações necessárias em um processador multiciclo simples para que se introduza a função de pipeline?
R - A adição de Registradores do Pipeline, que guardam os dados da transição de cada fase das cinco fases, denominados de (if/id), (id/exe), (exe/mem) e (mem/wb), que corresponde aos dados de cada transição. Apesar de o multiciclo já ter registradores adicionais ele não trabalha com a guarda dos dados de cada transição.

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: **1 - subi $t2,** $t2, 4
2 - lw $t1,** 0(**$t2)
3 - add $t3, $t1, $t4
4 - add $t4, $t3, $t3
5 - sw $t4,** 0(**$t2)
6 - beq $t2, $0, loop

2) 3) 4) Conflito de dados: o valor de t2 não terminou de ser calculado para ser usado no load,o mesmo para o t1.
3) Conflito estrutural: o add vai querer acessar a memória, mas o load acessa a memória duas vezes, então a instrução c só começa quando a b) terminar.
4) e 5) Conflito de dados pelo valor de t3 e t4.

Sem pipelining 38 ns
Com pipelining 33ns
Speedup 38/33 de 15%

4) 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

Existem 3 dependencias de dados reais

- **Div.d** e **sub.d**
- **Sub.d** e **s.d**
- **Add.d** e **mul.d**

Existe uma antidependencia entre sub.d e o add.d

- **Add** pode escrever em reg. Que **sub** lê.
- **War:** Uso de F5 pro **Sub.d**

Existe uma dependência de saída entre a sub.d e o Mul.d

- **Sub.d** pode termidar depois de **mul.d**
- **Waw:** Uso de **F4**

5) Em relação a memória cache. Um computador tem CPI 1 quando todos os acessos à memória acertam no cache. Loads e Stores totalizam 50% das instruções. Se a penalidade por miss é de 25 ciclos e o miss rate é 2%, qual o desempenho relativo se o computador acertar todos os acessos?

- Sempre acertando:

CPU execution time = (CPU clock cycles + Memory stall cycles) x Clock cycle
= (IC x CPI + 0) x Clock cycle
= IC x 1.0 x Clock cycle

- Situação real:

Memory stall cycles = IC x (Memory accesses / Instruction) x Miss rate x Miss penalty
= IC x (1 + 0.5) x 0.02 x 25
= IC x 0.75

CPU execution time$_{(cache)}$ = (IC x 1.0 + IC x 0.75) x Clock cycle
= 1.75 x IC x Clock cycle

- Resultado:

$$\frac{\text{CPU execution time}_{(cache)}}{\text{CPU execution time}} = \frac{1.75 \times IC \times \text{Clock cycle}}{1.0 \times IC \times \text{Clock cycle}}$$

$$= 1.75$$

6) Descreva os seguintes conceitos:

a) Write through : Um esquema em que as escritas sempre atualizam a cache e a memória, garantindo que os dados sejam sempre consistentes entre os dois.
PATTERSON, David. Organização e Projetos de Computadores: a interface hardware/software. 3 ed. Campus, 2005. Cap 7, pag 365.

b) Write back: Um esquema que manipula escritas atualizando valores apenas no bloco da cache e, depois, escrevendo o bloco modificado no nível inferior da hierarquia quando o bloco é substituído.
PATTERSON, David. Organização e Projetos de Computadores: a interface hardware/software. 3 ed. Campus, 2005. Cap 7, pag 365.

c) Localidade Temporal: Se um item é referenciado, ele tenderá a ser referenciado novamente em breve. Se você trouxe um livro para sua mesa para examiná-lo, é provável que precise examiná-lo novamente em breve.
PATTERSON, David. Organização e Projetos de Computadores: a interface hardware/software. 3 ed. Campus, 2005. Cap 7, pag 354.

d) Localidade Espacial: Se um item é referenciado, os itens cujos endereços estão próximos tenderão a ser referenciados em breve. Por exemplo, ao trazer o livro sobre os primeiros computadores ingleses para pesquisar sobre o EDSAC, você também percebeu que havia outro livro ao lado dele na estante sobre computadores mecânicos; então resolveu trazer também esse livro, no qual, mais tarde, encontrou algo útil. Os livros sobre o mesmo assunto são colocados juntos na biblioteca para aumentar a localidade espacial.
PATTERSON, David. Organização e Projetos de Computadores: a interface hardware/software. 3 ed. Campus, 2005. Cap 7, pag 354.