Lista de Exercícios – Cap. 4

1. [0.5] Explique o conceito de pipeline e descreva quais as vantagens e desvantagens deste tipo de implementação?
2. [1.0] Considere que os estágios individuais do caminho de dados têm as latências descritas na tabela 1. Também deve ser assumido que todas as instruções de uma determinada aplicação executadas pelo processador podem ser divididas nas classes conforme apresentado na tabela 2.

IF	ID	EX	MEM	WB
250ps	350ps	150ps	300ps	200ps

Tabela 1- Tempo dos estágios do pipeline

ALU/logic	Jal/branch	load	store
35%	20%	35%	10%

Tabela 2 – Frequência de classes de instruções

   a. Qual é o tempo de ciclo do clock em um processador com pipeline e sem pipeline (mono-ciclo)?
   b. Qual é a latência total de uma instrução LD em um processador com e sem pipeline?
   c. Se pudermos dividir um estágio do caminho de dados em pipeline em dois novos estágios, cada um com metade da latência do estágio original, qual estágio você dividiria e qual é o novo tempo de ciclo do clock do processador?
   d. Supondo que não haja stalls ou conflitos, qual é a utilização da memória de dados?
   e. Supondo que não haja stalls ou conflitos, qual é a utilização da porta de escrita do registrador da unidade de "Registradores"?

3. [1.0] Considere o programa descrito abaixo. O registrador x13

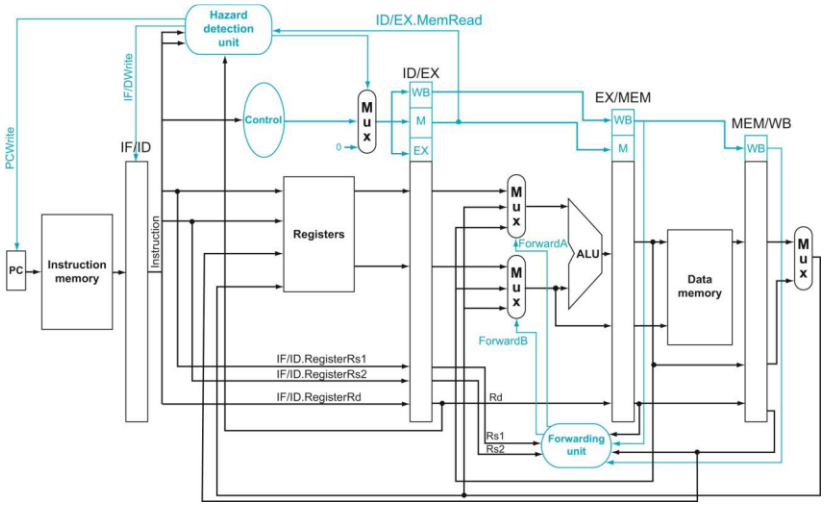
```
LOOP: ld x10, 0(x13)
      ld x11, 8(x13)
      add x12, x10, x11
      srli x12, x12, 2
      addi x13, x13, -8
      bnez x13, LOOP
```

   a. Considerando que os conflitos de dados e de controle foram resolvidos com a inserção de NOPs pelo compilador, qual o código será compilado e como fica o diagrama de execução de pipeline para as duas primeiras iterações deste loop.
   b. Considerando que o loop executa 16 iterações, qual o CPI do programa?
   c. Suponha que foi implementada no pipeline uma unidade de detecção de conflitos e uma unidade de encaminhamento, como fica o código gerado pelo compilador? Qual o novo CPI?
   d. Considerando o pipeline da letra c, suponha que a resolução do desvio foi adiantada para o estágio ID, qual o novo CPI?

4. [1.5] Considere a sequência de instruções abaixo que é executada em um pipeline de cinco estágios

```
sub x15, x12, x11
ld x13, 8 (x15)
ld x12, 0 (x15)
or x13, x15, x13
and x13, x13, x12
sd x13, 0 (x15)
```

   a. Calcule o tempo de execução considerando NOPs foram inseridos para garantir a execução correta.
   b. É possível reorganizar o código para minimizar o número de NOPs necessários?
   c. Se o processador tem uma unidade de adiantamento, mas a unidade de detecção de conflito não foi implementada, o que acontece quando o código original é executado?
   d. Se o processador tem uma unidade de adiantamento, especifique quais sinais são setados em cada ciclo nas unidades de adiantamento (forwarding) e de detecção de conflitos na Figura abaixo



Pipeline com encaminhamento (forwarding) e detecção de conflito.

5. {1,5] Considere a versão do pipeline do livro texto que não possui a técnica de *forwarding* (ou seja, o programador é responsável por inserir instruções NOP quando necessário). Suponha que (após otimizações pelo compilador) um programa com n instruções requer um adicional de 0,4 * n instruções NOP para resolver as dependências de dados.
   a. Considere que o tempo de ciclo do pipeline sem *forwarding* é de 250 ps. Suponha que a adição do hardware de *forwarding* reduzirá o número de NOPs de 0,4*n para 0,05*n, mas aumentará o tempo de ciclo para 300 ps. Qual o speed-up do pipeline com *forwarding* em comparação com o pipeline sem *forwarding*?
   b. Programas diferentes exigirão diferentes quantidades de NOPs. Quantos NOPs (como uma porcentagem do código instruções) podem permanecer no programa antes que programa seja executado mais lentamente no pipeline com *forwarding*?

c. Refaça a questão da letra b; no entanto, desta vez, usando x para representar o número de instruções NOP relativas a n (na letra b x era igual a 0,4). A resposta deverá ser com relação x.
d. Seria possível um programa com apenas 0,075*n NOPs executar mais rápido no pipeline com *forwarding*? Justifique.
e. No mínimo, quantos NOPs (em percentagem de instruções) deve um programa ter antes que ele possa executar mais rápido no pipeline com *forwarding* ?

6. [1.0] A importância de ter um bom preditor de desvio depende da frequência com que os desvios condicionais são executados. Junto com a precisão do preditor de desvio, este dado determinará durante quanto tempo o pipeline fica paralisado devido aos desvios incorretos. Neste exercício, suponha que a frequência de execução das classes de instruções seja a seguinte:

Tipo R	Beq/bne	jal	ld	sd
30%	35%	8%	2%	7%

Assuma que a precisão de alguns preditores de desvios para a aplicação é dada por:

Always Taken	Always not taken	2 Bits
42%	58%	87%

a. Os ciclos de paralisação devido aos desvios mal previstos aumentam o CPI. Qual é o CPI extra devido a desvios errados com a técnica "Always Taken"? Suponha que os resultados dos desvios sejam determinados no estágio de EX e aplicados no estágio MEM. Adicionalmente não há conflitos de dados e que nenhum slot de atraso é usado.
b. Calcule o CPI para o preditor "Always not taken".
c. Calcule o CPI para o preditor de 2 bits.
d. Com o preditor de 2 bits, que speed-up seria alcançado se pudéssemos converter metade das instruções de desvio para alguma instrução ALU? Suponha que as instruções preditas correta e incorretamente tenham a mesma chance de serem substituídas.
e. Algumas instruções de branch são muito mais previsíveis do que outras. Se sabemos que 70% de todas as instruções de desvio executadas são desvios de loop-back fáceis de prever e que são sempre previstos corretamente, qual é a precisão do preditor de 2 bits nos 30% restantes das instruções de desvio?

7. [2.5] Neste exercício, o desempenho de processadores de 1-issue e 2-issue serão comparados, levando em consideração as transformações do programa que podem ser feitas para otimizar a execução de um processador com 2-issue estático. As questões neste exercício referem-se ao código abaixo que calcula o loop dado por:

```
for (i = 0; i != j; i+ = 2) {
        c[i] = (d[i] + d[i+1])*8;
    }
```

```
        addi x12, x0, 0
        jal x0,EP
AGAIN:  slli x5, x12, 3
        add x6, x10, x5
        ld x7, 0 (x6)
        ld x25, 8 (x6)
        add x26, x7, x25
        slli x26,  x26, 3
        add x27, x11, x5
        sd x26, 0 (x27)
        addi x12, x12, 2
EP:     bne x12, x13, AGAIN
```

Com o uso dos seguintes registradores para armazenar as variáveis

i	j	End. d	End. c	Valores temporários
x12	x13	x10	x11	x5-x7, x25-x27

Suponha que o processador com static 2-issue tenha as seguintes propriedades:

1. Uma instrução deve ser uma operação de acesso à memória; a outra pode ser uma instrução aritmética / lógica ou um desvio.
2. O processador tem unidades de adiantamento e detecção de conflitos e o desvio é resolvido no estágio ID.
3. O processador tem uma previsão de desvio perfeita.
4. Duas instruções podem não ser despachadas juntas em um pacote se uma depender da outra.
5. Se um stall for necessário, ambas as instruções no pacote de issue devem parar.

Considerando as propriedades descritas acima responda às questões a seguir.

a. Desenhe um diagrama de pipeline mostrando como o código fornecido acima é executado no processador com 2-issue estático. Suponha que o loop termine após quatro iterações.
b. Qual é o speed-up obtido considerando um processador de 1-issue e um de 2-issue? (Suponha que o loop execute milhares de iterações.)
c. Desenrole o código de forma que cada iteração do loop desenrolado trate de duas iterações do loop original. Em seguida, reorganize / reescreva seu código desenrolado para obter melhor desempenho no processador de um issue. Você pode assumir que j é um múltiplo de 4.
d. Qual é o speed-up obtido considerando um processador de 1-issue e um de 2-issue considerando os códigos otimizados da letra (c)?
e. Repita os exercícios das letras (c) e (d), mas desta vez suponha que o processador 2-issue pode executar duas instruções aritméticas / lógicas juntas. (Em outras palavras, a primeira instrução em um pacote pode ser qualquer tipo de instrução, mas a segunda deve ser uma instrução aritmética ou lógica. Duas operações de memória não podem ser programadas ao mesmo tempo.)
f. Na sua opinião qual as vantagens e desvantagens de static multi-issue?

8. [1.0] Descreva a técnica de execução especulativa dinâmica com preditor de 2 bits. Quais as vantagens e desvantagens de se executar o código da questão anterior em um processador com execução especulativa dinâmica?