

Escola	EACH		TURMA		Nota do aluno na PROVA
Curso	Sistemas de Informação				
Disciplina	Arquitetura de Computador		Data da Prova	12/11/20	
Professor	Clodoaldo Aparecido de Moraes Lima				
Aluno					
No. USP					

1ª Questão) Um processador RISC é implementado em duas versões de organização síncrona: uma monociclo, em que cada instrução executa em exatamente um ciclo de relógio, e uma versão pipeline de 5 estágios. Os estágios da versão pipeline são: (1) busca de instrução, (2) busca de operandos, (3) execução da operação, (4) acesso à memória e (5) atualização do banco de registradores. A frequência máxima de operação das organizações foi calculada em 100 MHz para a versão monociclo e 400 MHz para a versão pipeline. Um programa X que executa 200 instruções é usado para comparar o desempenho das organizações. Das 200 instruções, apenas 40% fazem acesso à memória, enquanto as demais operam apenas sobre registradores internos da organização. Assuma, que o programa não apresenta nenhum conflito de dados ou de controle entre instruções que podem estar simultaneamente dentro do pipeline da segunda organização. Calcule o tempo de execução do programa X nas organizações monociclo e pipeline.

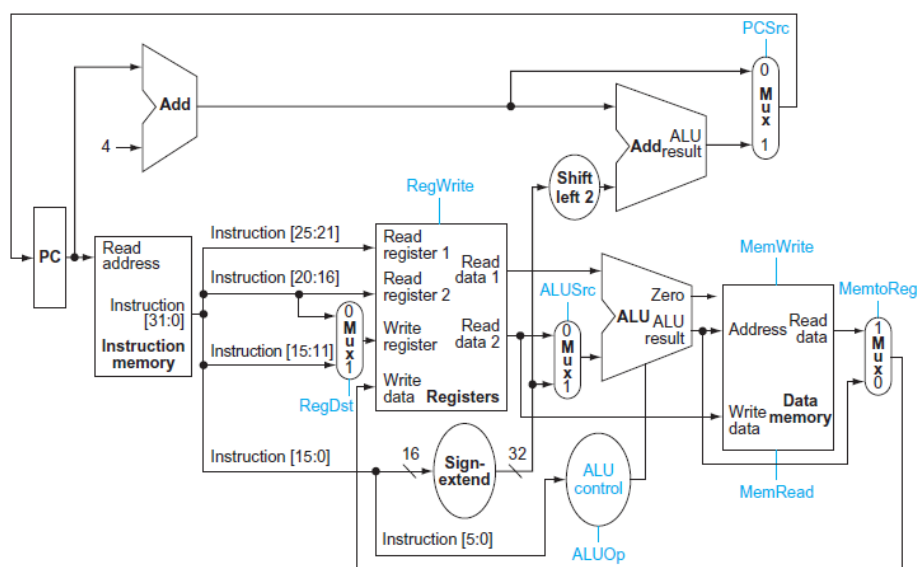
2ª Questão) Supondo que melhoramos uma máquina fazendo as instruções de *adição* serem executadas 2 vezes mais rápidas, e as instruções de *multiplicação* 4 vezes mais rápidas. Se o tempo de execução de certo *benchmark* antes do melhoramento é de 100s, sendo 20s em adição e 40s em multiplicação.

- calcular o *speedup*
- calcular o fator de melhoramento total

3ª Questão) Suponhamos que melhoramos uma máquina fazendo todas as instruções de ponto-flutuante serem executadas 5 vezes mais rápido.

- Se o tempo de execução de certo benchmark antes do melhoramento é de 10s, qual seria o speedup se metade dos 10s é dependida em instruções de ponto-flutuante?
- Estamos procurando um benchmark para testar a nova unidade de ponto-flutuante acima, e queremos que o benchmark todo mostre um speedup de 3. Um benchmark é executado em 100s, com o antigo hardware de ponto-flutuante. Quanto do tempo de execução as instruções de ponto-flutuante devem corresponder nesse programa para que possamos produzir o speedup desejado nesse benchmark?

4ª Questão) Apresente os valores dos sinais de controle *ALUSrc*, *ALUOp*, *MemRead*, *MemWrite*, *RegWrite*, *MemtoReg* e *RegDst*, na implementação da instrução *addi* no MIPS pipeline, indicando em que estágio cada um desses sinais são usados.



PROVA P1

5ª Questão) Por que a instrução `addi` não pode ser considerada uma instrução de formato R, com opcode igual 000000?

6ª Questão) Descreva detalhadamente o que cada código faz

<p>a)</p> <pre> .text main: li \$v0, 5 syscall move \$t0, \$v0 li \$v0, 5 syscall move \$t1, \$v0 bgt \$t0, \$t1, t0_bigger move \$t2, \$t1 b endif t0_bigger: move \$t2, \$t0 endif: move \$a0, \$t2 li \$v0, 1 syscall li \$v0, 10 syscall </pre>	<p>b)</p> <pre> .text .globl main main: li \$a0, 5 jal Funcao move \$s0, \$v0 li \$v0, 10 syscall Funcao: sub \$sp, \$sp, 4 sw \$ra, 0(\$sp) li \$t1, 1 slti \$t0, \$a0, 2 beq \$t0, \$zero, Calcula add \$v0, \$zero, \$zero beq \$a0, \$zero, Sai add \$v0, \$t1, \$zero Sai: lw \$ra, 0(\$sp) add \$sp, \$sp, 4 jr \$ra Calcula: add \$a1, \$a0, \$zero Loop: sub \$a1, \$a1, \$t1 jal Multiplica add \$a0, \$v0, \$zero bne \$a1, \$t1, Loop j Sai Multiplica: mult \$a0, \$a1 mflo \$v0 jr \$ra </pre>
---	---

7ª Questão) Considere o seguinte loop em MIPS

```

LOOP: slt $t2, $0, $t1
beq $t2, $0, DONE
subi $t1, $t1, 1
addi $s2, $s2, 2
j LOOP
DONE:

```

a) Assuma que o registrador `$t1` é inicializado com o valor 0. Qual é o valor no registrador `$s2` assumindo que `$s2` inicialmente possui valor zero.

b) Para o código escrito em assembly MIPS, escreva a rotina equivalente em C. Assuma que os registradores `$s1`, `$s2`, `$t1` e `$t2` são inteiros A, B, i e temp, respectivamente.

8ª Questão) Traduza o seguinte código em C para assembly MIPS. Use o número mínimo de instruções. Assuma que os valores de a, b, i e j estão armazenados nos registradores `$s0`, `$s1`, `$t0` e `$t1` respectivamente. Também, assumo que registrador `$s2` armazene o endereço base da vetor D

```

for(i=0; i<a; i++)
for(j=0; j<b; j++)
    D[4*j] = i + j;

```

PROVA P1

9ª Questão) Traduza o seguinte loop para C. Assuma que o inteiro i é armazenado no registrador \$t1, \$s2 armazena o inteiro chamado result, e \$s0 armazena o endereço base do inteiro MemArray

```
addi $t1, $0, $0
LOOP: lw $s1, 0($s0)
add $s2, $s2, $s1
addi $s0, $s0, 4
addi $t1, $t1, 1
slti $t2, $t1, 100
bne $t2, $s0, LOOP
```

10ª Questão) Os problemas neste exercício referem-se a seguinte sequência de instruções, e suponha que seja executada em um pipeline de 5 estágios:

```
add $5, $2, $1
lw $3, 4($5)
lw $2, 0($2)
or $3, $5, $3
sw $3, 0($5)
```

a) Se não houver encaminhamento ou detecção de conflito, insira nops para garantir a execução correta.

b) Use nops apenas quando um conflito não puder ser evitado alterando ou reorganizando essas instruções. Você pode assumir o registro \$7 pode ser usado para manter valores temporários em seu código modificado.

c) Se o processador tem encaminhamento, mas esquecemos de implementar a unidade de detecção de conflito, o que acontece quando este código é executado?

11ª Questão) Os problemas neste exercício referem-se a seguinte sequência de instruções, e suponha que seja executada em um pipeline de 5 estágios:

```
add $5, $2, $1
or $3, $5, $3
lw $3, 4($5)
lw $2, 0($2)
sw $3, 0($5)
```

a) Se não houver encaminhamento ou detecção de conflito, insira nops para garantir a execução correta.

b) Use nops apenas quando um conflito não puder ser evitado alterando ou reorganizando essas instruções. Você pode assumir o registro \$7 pode ser usado para manter valores temporários em seu código modificado.

c) Se o processador tem encaminhamento, mas esquecemos de implementar a unidade de detecção de conflito, o que acontece quando este código é executado?

12ª Questão) Assuma que o seguinte código é executado sobre um processador pipeline com 5 estágio, com adiantamento e um preditor de desvio (o qual assume que todo desvio para trás é tomado)

<p>Label1: lw \$1, 40(\$6)                    beq \$2, \$3, Label2 //tomado                    add \$1, \$6, \$4          Label2: beq \$1, \$2, Label1// não tomado                    sw \$2, 20(\$4)                    and \$1, \$1, \$4</p>
---

a) Desenhe o diagrama de execução para este código, assumindo que não há slots de atraso e o que desvio executa no estágio MEM

b) Qual é o speed-up alcançado ao mover a execução de desvio para o estágio DI. Assuma que a comparação no estágio DI não afeta o tempo de ciclo de clock.