UNIVERSIDADE DE SÃO PAULO

SEGUNDO SEMESTRE LETIVO DE 2022

PRIMEIRA PROVA

Escola	EACH	TURMA		Nota do aluno na PROVA
Curso	Sistemas de Informação			
Disciplina	Organização e Arquitetura de Computador II	Data da Prova	25/10/22	
Professor	Clodoaldo Aparecido de Moraes Lima			
Aluno				
No. USP				

1a Questão) (1 ponto)O fragmento de código abaixo processa um array de palavras e retorna 2 valores importantes em $v0 e $v1. Assuma que o array tem armazenado 5.000 palavras indexadas de 0 a 4.999, sabendo que o endereço base do array está armazenado em $a0 e o tamanho do array (5.000) em $a1. Descreva em uma frase o que faz o programa e o que é retornado em $v0 e $v1

[1]		addu $a1, $a1, $a1
[2]		addu $a1, $a1, $a1
[2]		addu $v0, $zero, $zero
[4]		addu $t0, $zero, $zero
[5]	outer:	addu $t4, $a0, $t0
[6]		lw $t4, 0($t4)
[7]		addu $t5, $zero, $zero
[8]		addu $t1, $zero, $zero
[9]	inner:	addu $t3, $a0, $t1
[10]		lw $t3, 0($t3)
[11]		bne $t3, $t4, skip
[12]		addiu $t5, $t5, 1
[13]	skip:	addiu $t1, $t1, 4
[14]		bne $t1, $a1, inner
[15]		slt $t2, $t5, $v0
[16]		bne $t2, $zero, next
[17]		addu $v0, $t5, $zero
[18]		addu $v1, $t4, $zero
[19]	next:	addiu $t0, $t0, 4
[20]		bne $t0, $a1, outer

2a Questão) (2.0 Pontos) Escreva um programa em linguagem de montagem (assembly language) do processador MIPS que processa um vetor de números inteiros (VET), transformando cada elemento do vetor em seu valor absoluto. Por exemplo, o valor absoluto do número -43 é +43 e do número +55 é +55. O programa deve obrigatoriamente chamar uma sub-rotina calc_abs que recebe como parâmetro um número e calcula o valor absoluto deste, retornando-o segundo as convenções do MIPS. O valor retornado pela sub-rotina deve ser armazenado, pelo programa principal, na mesma posição do array que continha o elemento original. Utilize a área de dados abaixo para escrever o seu programa.

```
.data
VET: .word 23 -43 55 -9 -7 21 -76 12 -45 -10
TAM: .word 10
```

3a Questão) (2 Pontos) Determine o número real de ciclos de clock para executar uma vez o trecho abaixo (do blez $t1,end até uma instrução que salte para trás ou até a última instrução do trecho ser executada, o que acontecer primeiro). Suponha máxima capacidade de resolução de conflitos de dados (isto inclui uma unidade de adiantamento capaz de adiantar dados da saída do terceiro estágio para a entrada do terceiro estágio, da saída do quarto estágio para a entrada do terceiro, e da

saída do quarto estágio para a entrada do quarto estágio). Assuma também que está disponível uma estrutura mestre-escravo para acesso ao banco de registradores e um preditor de saltos de 2 bits que inicialmente prevê salto não-realizado. O PC é escrito no quarto ciclo de relógio de cada instrução. Detalhe a execução no diagrama pipeline abaixo, indique todos os adiantamentos de dados que ocorrerem (se ocorrerem) e mostre as bolhas nas posições adequadas, caso estas existam.

BLEZ -- Branches if the register is less than or equal to zero

Os valores iniciais dos registradores pertinentes são: $t0=0x10010000, $t1=0x44 (=68 base 10), $t2=0, $t4=0, $t3=0x100100AA

4a Questão) (2,0 Pontos) Considere a seguinte sequência de instruções, executadas em um datapath com pipeline de 5
estágios:

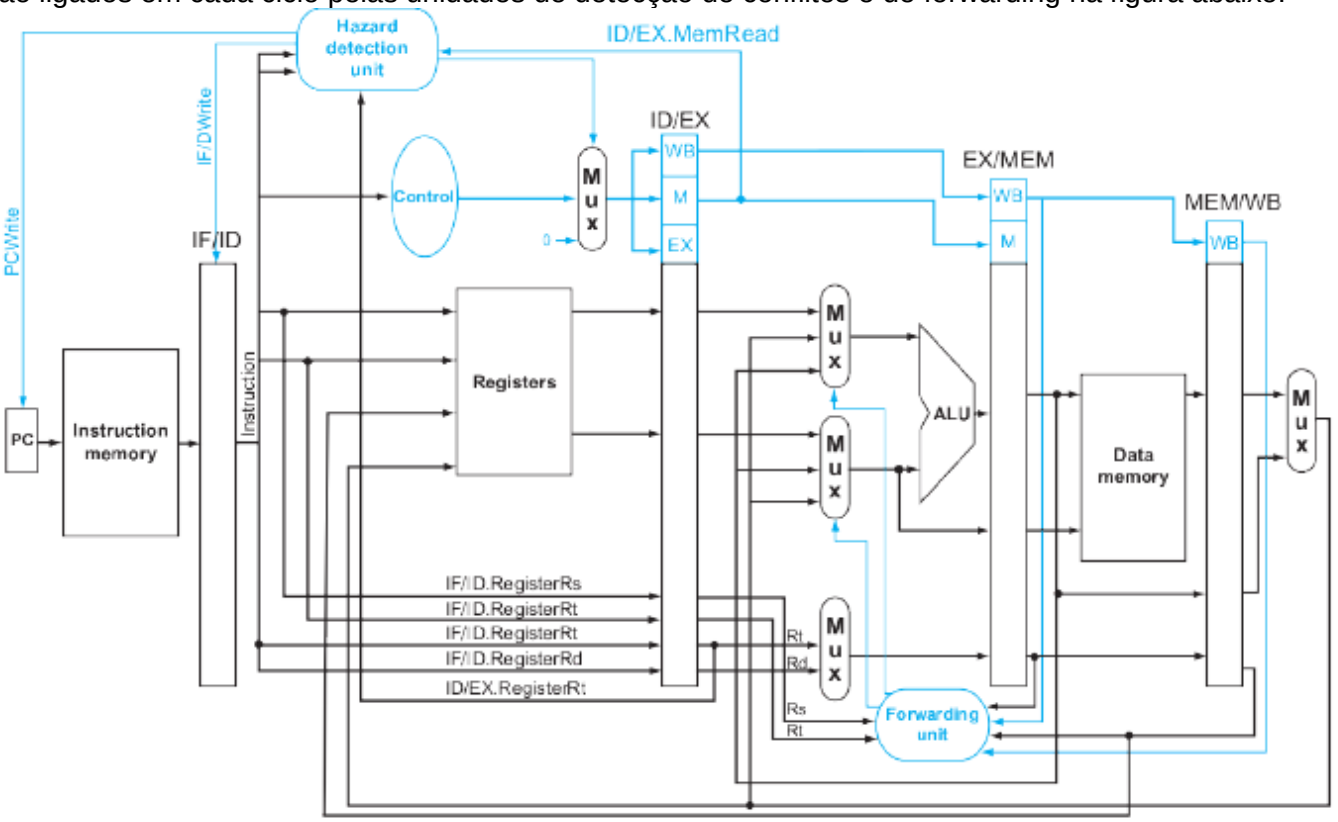
```
add $5, $2, $1
lw   $3, 4($5)
lw   $2, 0($2)
or   $3, $5, $3
addi  $4,$3,4
```

a) Na ausência de forwarding ou de detecção de conflito, insira nops para garantir que o código rode corretamente

b) Repita o item anterior usando nops somente quando um conflito não puder ser evitado pela mudança ou rearranjo dessas instruções. Assuma que o registrador $7 é usado para armazenar valores temporários no seu código modificado

c) Se o processador tiver forwarding, mas esquecermos de implementar a unidade de detecção de conflitos, o que ocorrerá quando este código for executado?

d) Na presença de forwarding, especifique, para os primeiros 5 ciclos da execução deste código, que sinais são ligados em cada ciclo pelas unidades de detecção de conflitos e de forwarding na figura abaixo:



UNIVERSIDADE DE SÃO PAULO

SEGUNDO SEMESTRE LETIVO DE 2022

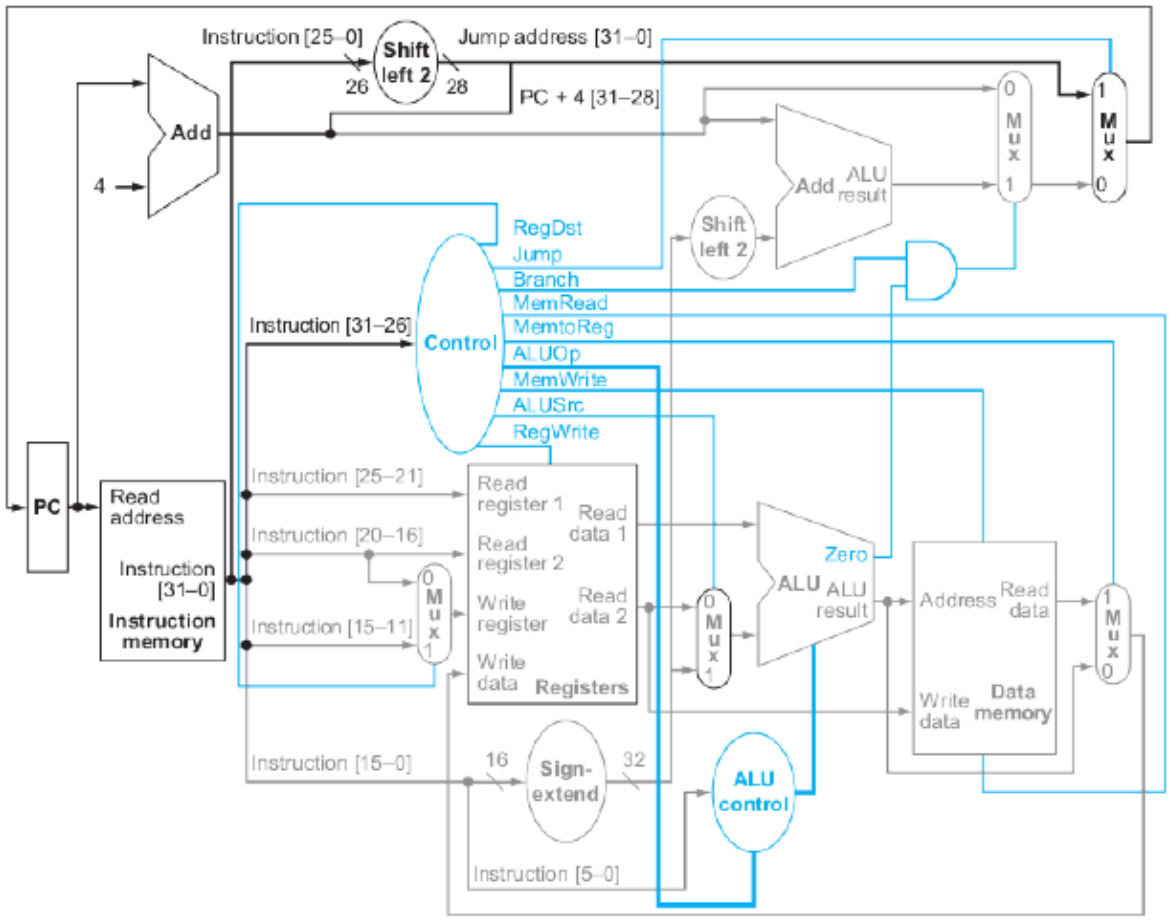
PRIMEIRA PROVA

Ciclo	PCWrite	IF/IDWrite	ID/EX	ALUin1	ALUin2
1					
2					
3					
4					
5					

d) Na ausência de forwarding, que novos sinais de entrada e saída precisaríamos ter na unidade de detecção de conflitos na figura acima? Usando esta sequência de instruções como exemplo, explique porque cada sinal é necessário.
(f) Para a nova unidade de detecção de conflitos acima, especifique que sinais de saída ela liga em cada um dos 5 primeiros ciclos, durante a execução desse código.

	Ciclo					Sinais		
	1	2	3	4	5	PCWrite	IF/IDWrite	ID/EX
Add $5,$2, $1								
lw $3,4($5)								
lw $2,0($2)								
or $3,$5,$3								
addi $4,$3,4								

5a Questão) (2,0 Pontos) Considere o datapath abaixo:



Agora suponha que, em um datapath de ciclo único, a seguinte instrução é trazida da memória: 10101100011000100000000000010100 (trata-se de um \sw rt, desl(rs)"). Assuma que a memória de dados está preenchida com zeros, e que os registradores do processador têm os seguintes valores no início do ciclo no qual a instrução acima é buscada:

r0	r1	r2	r3	r4	r5	r6	r8	r12	r31
0	–1	2	–3	–4	10	6	8	2	–16

a) Quais as saídas da extensão de sinal e da unidade de deslocamento de jumps (\Shift left 2" na figura) para essa instrução?
b) Quais os valores da entrada da unidade de controle da ALU para essa instrução?
c) Qual o novo endereço do PC após essa instrução ser executada? Mostre (na figura) o caminho que leva à definição desse valor.
d) Para cada MUX, mostre os valores de suas saídas durante a execução desta instrução com estes valores de registradores.
e) Para a ALU e as duas unidades de soma, quais são os valores de suas entradas de dados?
f) Quais os valores de todas as entradas para a unidade de registradores?

6a Questão) Limitando nossa atenção a 8 instruções: lw, sw, add, sub, and, or, slt e beq, qual o tempo médio entre 2 instruções em uma implementação de ciclo único, em que todas as instruções ocupam um único ciclo de clock, e uma implementação de pipeline, em que cada estágio ocupa um ciclo de clock? O tempo de cada instrução, tanto total quanto a cada estágio, é:

Instruction class	Instruction fetch	Register read	ALU operation	Data access	Register write	Total time
Load word (`lw`)	200 ps	100 ps	200 ps	200 ps	100 ps	800 ps
Store word (`sw`)	200 ps	100 ps	200 ps	200 ps		700 ps
R-format (`add`, `sub`, `AND`, `OR`, `slt`)	200 ps	100 ps	200 ps		100 ps	600 ps
Branch (`beq`)	200 ps	100 ps	200 ps			500 ps