

Universidade de São Paulo

Escola de Artes, Ciências e Humanidades

Docente: Prof. Dr. Clodoaldo A M Lima.

Discente: _____ No. USP: _____

Primeira Prova de Arquitetura de Computador

1ª Questão) (2,5 pontos) Apresente a implementação em linguagem de máquina dos códigos abaixo

Enquanto i<100	int fact(int n)
v[i+1] ← v[i] +1;	{
se v[i+1] <> 10	if (n < 1)
a++;	return 1;
fim se;	else
b ← c*2 -d;	return ((n+1)*fact(n-1));
fim enquanto	}

* função fact passagem de parâmetros pelo registrado \$a0

* a, b, c, d, i estão nos registrados \$t0, \$t1, \$t2, \$t3 e \$t4

Valor 1.0 Ponto

```
Loop:  slti  $t5, $t4, 100
      beq  $t5, $zero, FIM
      sll  $t6, $t4, 2      #multiplica i por 4
      add  $t6, $t6, $s0    # soma a base
      lw   $s1, 0($t6)     #carrega v[i]
      addi $s1, $s1, 1     #v[i]+1
      addi $t7, $t4, 1     #i=i+1
      slti $t7, $t7, 2     #multiplica i+1 por 4
      add  $t7, $t7, $s0    #soma a base
      sw   $s1, 0($t7)
      addi $t8, $zero, 10
      beq  $s1, $t8, FSE   #IF
      addi $t0, $t0, 1     #a = a+1
FSE:   addi $t4, $t4, 1     # i = i + 1
      multi $t6, $t2, 2
      sub  $t1, $t6, $t3
      j    Loop
```

FIM:

Esqueceu de multiplicar indice por 4

Cada instrução – 0.05

Valor 1.0 Ponto

fact:

```
addi $sp, $sp, -12
sw $ra, 8($sp)
sw $fp, 4($sp)
sw $ra, 0($sp)
move $fp, $sp
slti $t0, $a0, 1
beq $t0, $zero, L1
```

```

    addi $v0, zero, 1
    addi $sp, $sp, 12
    jr $ra
L1:
    addi $a0, $s0, -1
    jal fact
    lw $a0, 0($sp)
    lw $fp, 4($sp)
    lw $ra 8($sp)
    addi $sp, $sp, 12
    mult $va0, $a0, $v0
    jr $ra

```

2ª Questão) (2,0 pontos) As arquiteturas de uso geral atuais são normalmente referenciadas como máquinas von Neumann. Quais as principais características dessa arquitetura e sua principal limitação?

Características – 1.0 ponto

Conceito de programa armazenado; 0.50

- *Dados e instruções armazenados em uma única memória de leitura e escrita.*

Endereçamento da memória por posição e não pelo tipo; 0.50

Execução sequencial de instruções; e

Único caminho entre memória e CPU.

Limitação – 1.0 ponto

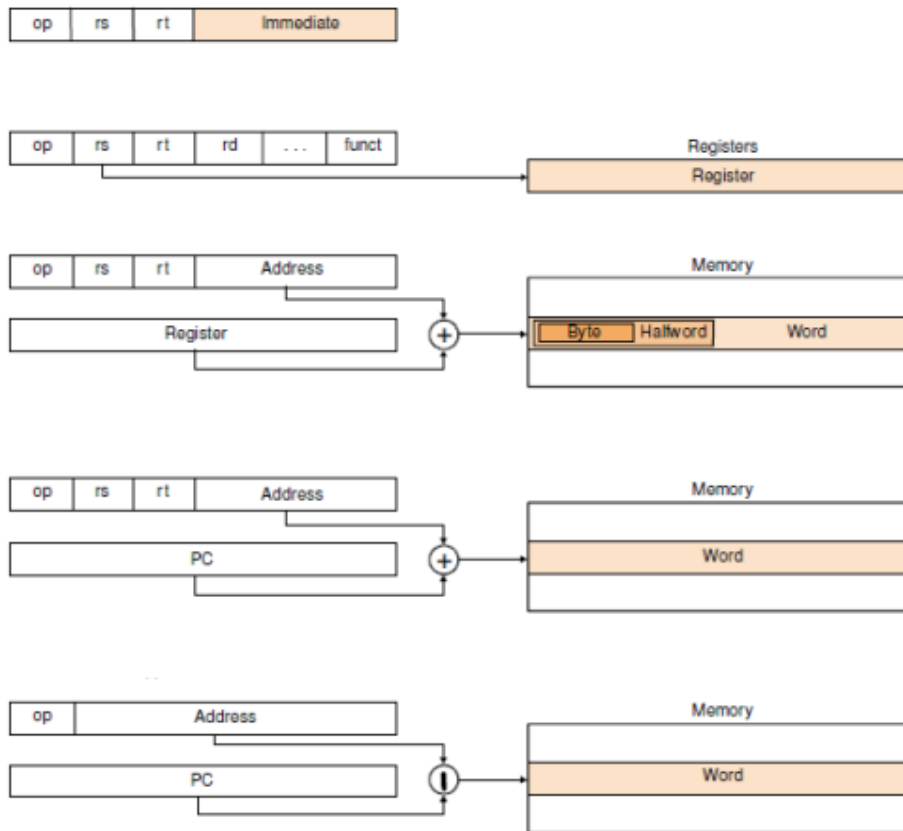
Tráfego intenso no barramento do sistema

- *Principal rota de informação: CPU e memória (ponto crítico)*
- *Constante fluxo de dados e instruções*

Gera desperdício de tempo (CPU em espera)

Agrava-se gradativamente pelo aumento do gap de velocidade entre a memória e a CPU

3ª Questão) (2,0 pontos) Explique como funcionam as 5 formas de endereçamento básicas da arquitetura MIPS mostradas. Compare as em termos de complexidade de execução da busca/utilização dos operandos:



Cada item – 0.4

- Endereçamento imediato: o operando é uma constante dentro da própria instrução. Operações aritméticas e de comparação: ADDI R1, R2, #A*
- Endereçamento a registrador: O operando está em um registrador e a instrução contém o número do registrador. ADD R3, R3, R7*
- Endereçamento via registrador-base ou via deslocamento: operando está no endereço de memória obtido com a soma do conteúdo do registrador-base com a constante armazenada na instrução (Const (\$reg)). Instruções de acesso à memória*
- Endereçamento relativo ao PC: operando está no endereço formado pela soma do conteúdo do PC com a constante obtida na própria instrução. Instrução branch. o número de instruções a serem puladas a partir da instrução é especificado na instrução*
- Endereçamento pseudo-direto: o endereço de desvio é formado pela concatenação dos 28 bits (26 bits estendido) obtido da instrução com os bits mais significativos do PC*

4ª Questão) (1,5 ponto) Escreva os códigos em linguagem de máquina para computar a expressão $X = (A + B \times C) / (A - B \times C)$ para as duas arquiteturas com instruções em linguagem de máquina disponíveis e formas de endereçamento para operações aritméticas conforme a tabela a seguir. Em seguida, determine o

número de bytes de instruções buscados na memória e o número de bytes de dados trocados entre a memória e a CPU.

Forma de Endereçamento	
Registrador-Memória	Registrador
Load R1, M	Load R1, M
Store R1, M	Store R1, M
Add R3, R2, M	Add R3, R2, R1
Sub R3, R2, M	Sub R3, R2, R1
Mult R3, R2, M	Mult R3, R2, R1
Div R3, R2, M	Div R3, R2, R1

OBSERVAÇÕES:

1. Considere possíveis **otimizações** no código
2. M é um endereço de memória 16 bits
3. Ri é um registrador de 4 bits
4. Os *opcodes* possuem 8 bits
5. As instruções devem ter **comprimentos múltiplos de 4 bits**
6. Note que a linguagem **não é MIPS**

Tabela REG – REG → 0.5 ponto

Buscado – 0.15

Trocado – 0.1

Tabela REG – MEM → 0.5 ponto

Buscado – 0.15

Trocado – 0.1

Considerando Add R1, R2, E # E ← R1+R2

Assembly REG-MEM	Significado	Bits Instrução	Bits Dados
Load R1, B	R1 = B	28	16
Load R2, C	R2 = C	28	16
Mult R1, R2, D	D = B x C	32	16
Load R1, A	R1 = A	28	16
Load R2, D	R2 = B x C	28	16
Add R1, R2, E	E = (A+BxC)	32	16
Sub R1, R2, F	F = (A-BxC)	32	16
Load R1, E	R1 = (A+BxC)	28	16
Load R2, F	R2 = (A-BxC)	32	16
Div R1, R2, X	X= (A+BxC)/ (A-BxC)	32	16

Considerando Add R1, R2, E # R1 ← R2+E

Assembly REG-MEM	Significado	Bits Instrução	Bits Dados
Load R1, B	R1 = B	28	16
Mult R1, R1, C	R1 = R1 x C = B x C	32	16
Store R1, D	D = R1 = B x C	28	16
Load R1, A	R2 = A	28	16
Add R2, R1, D	R2 = A + BxC	32	16
Sub R3, R1, D	R3= A - B*C	32	16
Store R3, D	D =R3 = A - Bx C	28	16
Div R3, R2, D	R3 =(A+BxC)/(A-BxC)	32	16
Store R3, X	X = (A+BxC)/(A-BxC)	28	16

Considerando Add R1, R2, R3 # R1 ← R1+ R3

Assembly REG-REG	Significado	Bits Instrução	Bits Dados
------------------	-------------	----------------	------------

Load R1, B	$R1 = B$	28	16
Load R2, C	$R2 = C$	28	16
Load R3, A	$R3 = A$	28	16
Mult R4, R1, R2	$R4 = (B \times C)$	20	0
Add R5, R3, R4	$R5 = (A + B \times C)$	20	0
Sub R4, R3, R4	$R4 = (A - B \times C)$	20	0
Div R3, R5, R4	$R3 = (A + B \times C) / (B - A \times C)$	20	0
Store R3, X	$X = (A + B \times C) / (D - E \times F)$	28	16

Duas instruções de multiplicação – descontar 0.1

Não atribui os valores a M (A, B, C) – descontar 0.1

Usar registradores R1A – descontar 0.05

Esqueceu uma instrução – descontar 0.1

Instrução errada – descontar 0.1

Troca entre Bits Instrução e Bits Dados – considera apenas 0.1 (se estiver tudo certo)

5ª Questão) (2,0 pontos) Considere a organização do bloco de dados multiciclo abaixo, que acomoda a execução de um subconjunto da arquitetura do conjunto de instruções do processador MIPS. Considere também as modificações que foram necessárias realizar nesta. Em seguida, responda às questões abaixo.

a) Marcar no desenho abaixo e/ou descrever todos os caminhos do bloco de dados efetivamente usados pela instrução BEQ Rs, Rt, Rótulo e JUMP Rótulo. Isto significa marcar e/ou descrever em texto todos os caminhos por onde passa informação útil relevante à execução da instrução, ou seja, os dados e endereços que esta realmente necessita manipular.

Beq Rs, Rt

Primeiro ciclo a instrução é buscada na memória e o PC é incrementado, o sinal FontePC é setado igual a 0. No segundo ciclo, a instrução é decodificada, o sinal de EscReg e RegDst é setado igual a 1 e X respectivamente. No terceiro ciclo, a subtração entre o conteúdo dos registradores Rs, Rt é realizada e os sinais de controle ULAOp e ULAFonte é setado igual a 01 a 0, respectivamente. Se o conteúdo destes registradores são iguais, a ULA gera um sinal igual a 1. Este sinal deve entrar numa porta and juntamente o sinal de controle (branch) vindo da unidade de controle. A saída da porta and é usada para setar multiplexador, tal que a saída deste seja a soma do PC+4 com os 16 bits da instrução estendido. Este valor é usado para atualizar o PC. Os outros sinais de controle (EscMem, LerMem, MemParaReg) são setados para 0.

JUMP Rótulo

Primeiro ciclo a instrução é buscada na memória e o PC é incrementado, o sinal FontePC é setado igual a 0. No segundo ciclo, a instrução é decodificada, o sinal de EscReg e RegDst é setado igual a 1 e X respectivamente. No terceiro ciclo, o endereço contendo 26 bits deslocado de 2 bits é concatenado com os bits mais significativos do PC. Os sinais de controle

ULAOp e ULAFonte pode ser X, já os sinais de controle (EscMem, LerMem, MemParaReg) devem ser setados para 0.

b) Diga qual operação é executada pela unidade lógica-aritmética (ALU) no terceiro ciclo de relógio da instrução BEQ Rs, Rt, Rótulo e JUMP Rótulo, justificando sua resposta.

Como já descrito no item anterior a ULA faz uma soma de suas entradas op1 e op2, para obter o endereço para onde saltar, caso o salto seja executado.

c) Qual a função da ALU na execução das instruções em cada ciclo das instruções na organização multiciclo.

Primeiro ciclo - Incrementa o PC

Segundo ciclo - Cálculo o endereço de destino do desvio

Terceiro ciclo - depende da instrução

Referência à memória - calcula o endereço de memória

Instrução R - realiza a operação especificada pelo código de função com dois valores lidos do banco de registradores

Desvio condicional - realiza a comparação de igualdade entre os dois registradores lidos na etapa anterior

JUMP Rótulo - não realiza nenhuma operação

Quarto ciclo - Acesso a memória ou conclusão da instrução tipo R

Quinto ciclo - Conclusão da leitura da memória

d) Explique detalhadamente como funciona uma unidade de controle microprogramada

É uma técnica de implementação de controladores lógicos que tem como objetivo administrar o correto funcionamento das atividades dentro e fora da CPU. Segue a figura que corresponde a implementação de um controle microprogramado. Basta explicar como funciona.

