13/12/2017

UFSM
Universidade Federal de Santa Maria

Departamento de Eletrônica e Computação
ELC 1011– Organização de Computadores

Nome:_____

4ª Prova (Gabarito)

1 – (3,0 pontos, 30 min.) Converta a instrução em linguagem de montagem **lw $t4, 16 ($t5)** para linguagem de máquina e explique detalhadamente, com a ajuda da figura 1A e as tabelas 7 e 8, como esta instrução é executada pelo processador monociclo. Apresente na figura, os sinais nos circuitos de controle e controle da UAL.

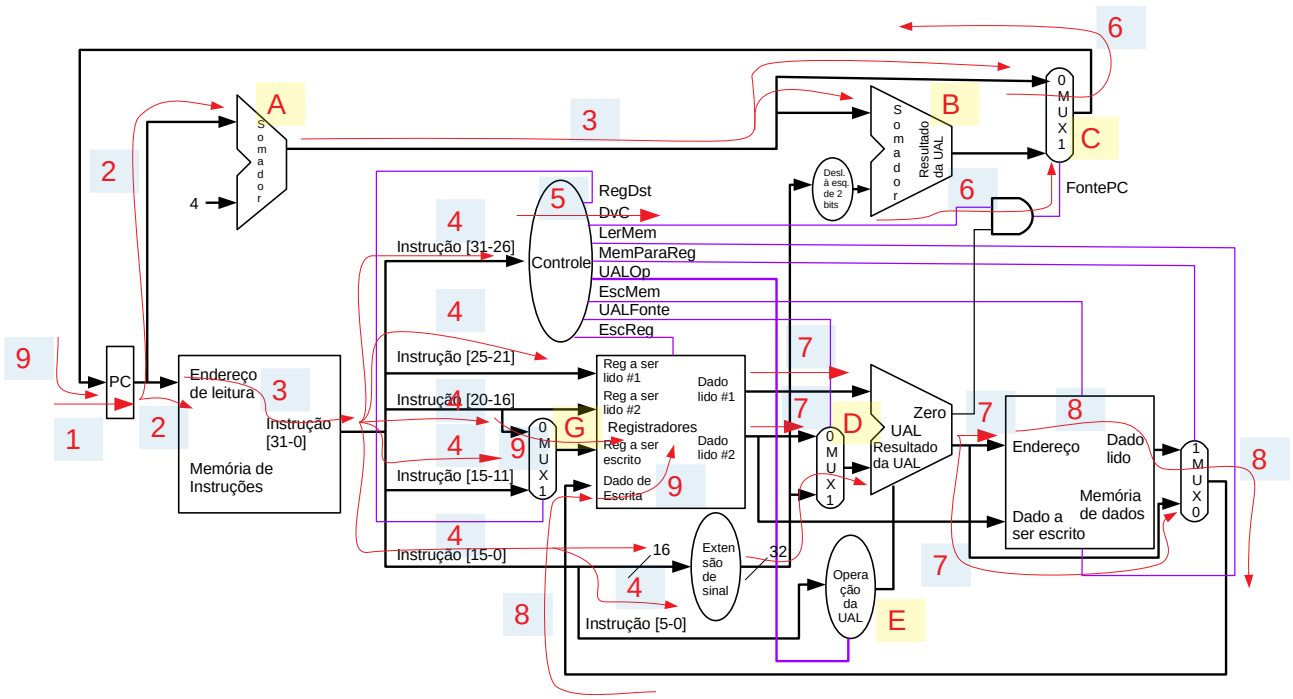

Figura 1A – Caminho de dados e de controle de um processador monociclo.

Solução:

Encontramos a instrução em linguagem de máquina. Da tabela 2, verificamos que é uma instrução tipo I, com o opcode 0x23 e os registradores rs = $t5 = $13 e rt = $t4 = $t12. Os campos da instrução tipo I são apresentados na figura 2. Temmos os seguintes valores para os campos:
opcode = 0x23 = 100011
rs = $13 = 01101
rt = $12 = 01100
valor imediato = 16 = 0000 0000 0001 0000

Juntamos os campos e encontramos a instrução em linguagem de máquina:

100011 01101 01100 0000000000010000 = 1000 1101 1010 1100 0000 0000 0001 0000 = 0x8DAC0010.

A seguir descrevemos a leitura, decodificação e execução desta instrução. Os passos descritos iniciam em uma borda de subida do sinal de relógio. A instrução é lida, decodificada e executada no intervalo de um

ciclo de relógio. Esta instrução termina na próxima borda de subida do relógio, momento em que uma nova instrução começa a ser processada.

1. Na borda de subida do sinal de relógio o novo valor de PC é registrado. PC contém o endereço da próxima instrução que será processada.
2. Um pouco após a borda de subida do sinal de relógio, o valor de PC aparece na saída do registrador, na entrada de endereços da memória de instruções e na entrada do somador indicado com um A na figura 1A.
3. A memória de instruções é lida e na saída de dados temos a instrução associada com este endereço. Neste exemplo, o valor na saída de dados é 0x8DAC0010, correspondente a instrução lw $t4, 16($t5). Neste tempo, temos o valor de PC+4 na saída do somador A. Este valor se propaga até o somador indicado por B e o multiplexador indicado por C.
4. Os campos da instrução são separados: Instrução[31-26] = opcode = 0x23, instrução[25-21] = rs = $13 = $t5, instrução[20-16] = rt = $12 = $t4, instrução[15-11] = rd = 00000 = $zero, Instrução[15-0] = 0x0010 = 16 e instrução[5-0] = funct = 0x10.
5. O subsistema de controle recebe o opcode e gera os sinais de controle. Na tabela 7, temos os sinais gerados pelo opcode da instrução: RegDst = 0, UalFonte=1, MemParaReg = 1, EscReg = 1, LerMem = 1, EscMem = 0, DvC = 0, UALOp1 = 0 e UALOp0 = 0.
6. A saída da porta AND, abaixo do multiplexador C, é igual a 0 porque o sinal DvC = 0. A saída 0 desta porta seleciona a entrada 0 do multiplexador marcado como C. Esta entrada possui o valor PC + 4. Este valor vai até a entrada do registrador PC. Na próxima borda de subida o valor PC + 4, correspondente ao endereço da próxima instrução, será registrado e a próxima instrução executada.
7. Na saída Dado lido #1 temos o conteúdo do registrador rs = $t5. Nas entradas do multiplexador marcado por D, temos o valor do registrador rt = $t4 e o valor imediato estendido (os 16 bits são estendidos para 32 bits). O sinal UALFonte = 1. A saída do multiplexador será igual ao valor extendido 16. Este valor é uma das entradas da UAL (Unidade Aritmética e Lógica). O circuito que gera a operação da UAL (marcado como E) recebe o campo funct = 0x10 e o sinal UALOp = 00 do subsistema de controle. Da tabela 8, vemos que para este sinal em UALOp, a saída da operação da UAL será 010 = soma, independente do campo funct (instrução[5-0]). A UAL realiza a soma das suas entradas: o valor do registrador rs = $t5 e o valor imediato estendido 16. A saída da UAL ($t5 + 16) aparece na entrada de endereço da memória de dados e na entrada 0 do mulitplexador na saída da memória de dados, marcado como F.
8. Na memória de dados temos os sinais de controle EscMem = 0 e LerMem = 1. O endereço $t5+16 é lido. O conteúdo deste endereço aparece na entrada 1 do multiplexador F. O sinal MemParaReg = 1. A entrada 1, com o conteúdo da memória do endereço $t5 + 16 aparece na saída do multiplexador F e propaga-se até a entrada Dado de escrita do banco de registradores.
9. O sinal de controle RegDst = 0. A entrada 0 com o endereço de rt = $t4 passa a saída do multiplexador. O sinal EscReg = 1, permitindo a escrita no banco de registradores. Na próxima borda de subida do sinal de relógio, o valor de Dado de escrita é escrito no registrador rt = $t4 e o processando da instrução é completada. A seguir, a instrução do endereço PC+4 da memória de instruções começa a ser processada.

2 – (3,0 pontos, 30 min.) Converta a instrução em linguagem de montagem **add $t4, $a1, $v0** para linguagem de máquina e explique detalhadamente, com a ajuda da figura 1B e as tabelas 7 e 8, como esta instrução é executada pelo processador monociclo. Apresente na figura, os sinais nos circuitos de controle e controle da UAL.

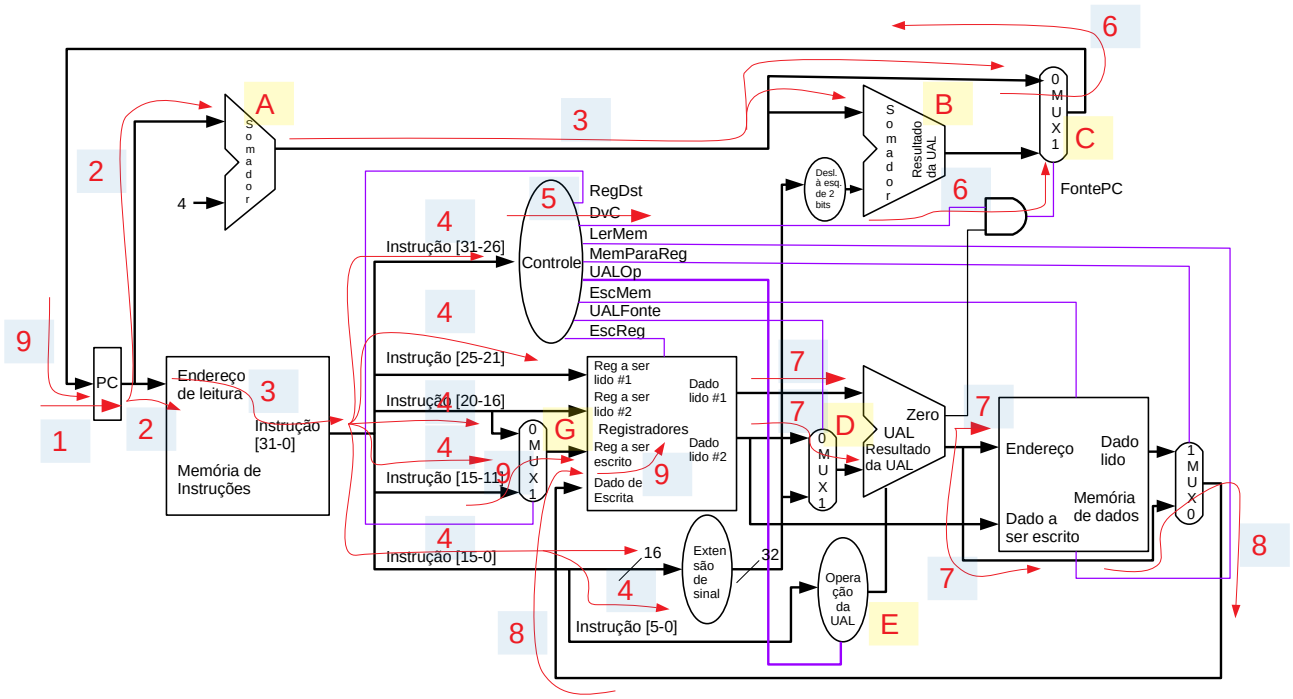

Figura 1B – Caminho de dados e de controle de um processador monociclo.

Solução:

Da tabela 2, temos que a instrução add $t4, $a1, $v0 é uma instrução tipo R (add rd, rs, rt), com o campo funct = 0x20. Da figura 1, encontramos os campos da instrução:
Opcode = tipo R = 0 = 000000
rd = $t4 = $12 = 01100
rs = $a1 = $5 = 00101
rt = $v0 = $2 = 00010
shamt = 0 (não temos deslocamentos) = 00000
funct = 0x20 = 100000

Juntamos os campos e construímos a instrução em linguagem de máquina:
000000 00101 00010 01100 00000 100000 = 0000 0000 1010 0010 0110 0000 0010 0000 = 0x00A26020.

A sguir iremos descrecer a execução, decodificação e execução desta instrução, usando a figura 1B e as tabelas 7 e 8.

1. Na borda de subida do sinal de relógio o novo valor de PC é registrado. PC contém o endereço da próxima instrução que será processada.
2. Um pouco após a borda de subida do sinal de relógio, o valor de PC aparece na saída do registrador, na entrada de endereços da memória de instruções e na entrada do somador indicado com um A na figura 1A.
3. A memória de instruções é lida e na saída de dados temos a instrução associada com este endereço. Neste exemplo, o valor na saída de dados é 0x00A26020, correspondente a instrução add $t4, $a1, $v0. Neste tempo, temos o valor de PC+4 na saída do somador A. Este valor se propaga até o somador indicado por B e o multiplexador indicado por C.
4. Os campos da instrução são separados: Instrução[31-26] = opcode = 000000, instrução[25-21] = rs = 00101 = $5 = $a1, instrução[20-16] = rt = 00010 = $2 = $v0, instrução[15-11] = rd = 01100 = $t12 = $t4, Instrução[15-0] = 0110 0000 0010 0000 = 0x6020 e instrução[5-0] = funct = 0x20.
5. O subsistema de controle recebe o opcode e gera os sinais de controle. Na tabela 7, temos os sinais gerados pelo opcode da instrução: RegDst = 1, UalFonte=0, MemParaReg = 0, EscReg = 1, LerMem = 0, EscMem = 0, DvC = 0, UALOp1 = 1 e UALOp0 = 0.
6. A saída da porta AND, abaixo do multiplexador C, é igual a 0 porque o sinal DvC = 0. A saída 0 desta

porta seleciona a entrada 0 do multiplexador marcado como C. Esta entrada possui o valor PC + 4. Este valor vai até a entrada do registrador PC. Na próxima borda de subida o valor PC + 4, correspondente ao endereço da próxima instrução, será registrado e a próxima instrução executada.
7. Na entrada da UAL temos o Dado lido #1 = rs = $a1 e a saída do multiplexador (multiplexador D). UALFonte = 0, sendo selecionada para a outra entrada da UAL o valor Dado lido #2 = rt = $v0. Da tabela 8, verificamos o código de operação enviado para a UAL. Como UALOp = 10 e o campo funct = 0x20 = 100000, o código de operação será uma soma. A soma do conteúdo dos registradores rs = $a1 e rt=$v0 é realizada pela UAL.
8. Os sinais de leitura da memória de dados LerMem = 0 e escrita da memória de dados EscMem = 0. Nenhuma operação sobre a memória de dados será realizada. O multiplexador F, recebe o sinal de controle MemParaReg = 0. A saída da UAL com $a1+$v0 aparece na saída deste multiplexador. O sinal se propaga até a entrada Dado de escrita do banco de registradores.
9. Na próxima borda de subida do sinal de relógio, a entrada Dado de escrita = $a1 + $v0 será escrito no banco de registradores, uma vez que o sinal EscReg = 1. A saída do multiplexador G será igual ao endereço do registrador rd = $t4, pois o sinal RegDst = 1. Este endereço aparece em Reg a ser escrito. Na borda de subida do sinal de relógio a soma $a1 + $v0 será escrita no registrador $t4.

3 – (4,0 pontos, 30 min.) Escreva um programa em linguagem de montagem para o processador MIPS, para calcular o fatorial de um número em ponto flutuante x (precisão dupla)..

$$fatorial(x) = \prod_{1}^{x} = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot x \tag{1}$$

Solução:

```
# Programa que calcula o fatorial de um número x
# O programa não verifica se x é um número inteiro (na representação
# em ponto flutuante!) ou se x é um número igual ou maior que zero.
# O maior valor para x é 170.

# segmento de texto - programa
.text
    # $f0 <- x
    # $f2 <- fatorial(x), o valor da função fatorial
    # $f4 <- constante 0
    # $f6 <- constante 1
    # as constantes poderiam estar armazenadas no segmento de dados
    # $f12 <- variável tmp (com o valor x, x-1, x-2, ..., 0)

    # carrega da memória a variável x
    la      $t0, variavel_x
    ldc1    $f0, 0($t0)    # $f0  <- x
    mov.d   $f12, $f0      # tmp = $f12 <- x
    # carrega a constante 0 para $f4
    li      $t0, 0         # $t0 <- 0
    mtc1    $t0, $f4       # $f4 <- $t0
    cvt.d.w $f4, $f4       # $f4 <- 0
    # carrega a constante 1 para $f6
    li      $t0,1          # $t0 <- 1
    mtc1    $t0, $f6       # $f6 <- $t0
    cvt.d.w  $f6, $f6      # $f6 <- 1
    # 0 valor inicial para o fatorial é 1
    # Desta forma, o calculo de 0! estará correto
    mov.d   $f2, $f6       # $f2 <- 1
    # cálculo do fatorial
calcula_fatorial:
    c.lt.d    $f4, $f12    # tmp > 0 ?
    bc1f termina_calculo_fatorial # se falso termina as iterações
    mul.d     $f2, $f12, $f2  # fatorial(x) = tmp *fatorial(x)
    sub.d     $f12, $f12, $f6 # tmp <- tmp - 1
    j calcula_fatorial
termina_calculo_fatorial:
    # $f12 possui o valor de fatorial(x)
fim_programa:
    li      $a0, 0         # código de saída = 0, saída normal
    li      $v0, 17        # serviço exit 2
    syscall

# segmento de dados na memória
.data

    variavel_x:  .double 20.0 # variável x, com um valor para teste
```

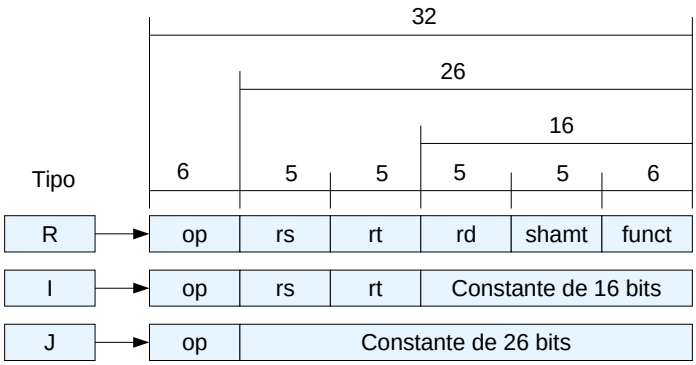

Figura 2 – Ilustração dos campos das instruções R, I e J de um processador MIPS.

Tabela 1 - Exemplos de instruções tipo R (opcode = 000000)

Instrução	Campo funct
add rd, rs, rt	0x20
addu rd, rs, rt	0x21
and rd, rs, rt	0x24
nor rd, rs, rt	0x27
sll rd, rt, shamt	0x00
sllv rd, rt, rs	0x04
sra rd, rt, shamt	0x03
sub rd, rs, rt	0x22
subu rd, rs, rt	0x23
slt rd, rs, rt	0x2A
sltu rd, rs, rt	0x2B

Tabela 2 - Exemplo de instruções tipo I

Instrução	Opcode
addi rt, rs, imediato	0x08
xori rt, rs, imediato	0x0E
slti rt, rs, imediato	0x0A
beq rs, rt, label	0x04
lb rt, imediato(rs)	0x20
lw rt, imediato(rs)	0x23
sb rt, imediato(rs)	0x28
sw rt, imediato(rs)	0x2b

Tabela 3 - Exemplo de instruções tipo J

Instrução	Opcode
j endereço-alvo	0x02
jal endereço-alvo	0x03

Tabela 4 – Nome e número dos registradores do MIPS.

Nome	**número**		**Nome**	**Número**
$zero	0		$t8 a $t9	24 a 25
$at	1		$k0 a $k1	26 a 27
$v0 a $v1	2 a 3		$gp	28
$a0 a $a3	4 a 7		$sp	29
$t0 a $t7	8 a 15		$fp	30
$s0 a $s7	16 a 23		$ra	31

Tabela 5 – Instruções em ponto flutuante

	Precisão	
	Simples	Dupla
Adição	add.s	add.d
Subtração	sub.s	sub.d
Multiplicação	mul.s	mul.d
divisão	div.s	div.d

Armazenamento → lwc1 ldc1 / swc1 ldc1

```
.text
    lwc1 $f1, -100($t2)
    swc1 $f1, -100($t2)
```

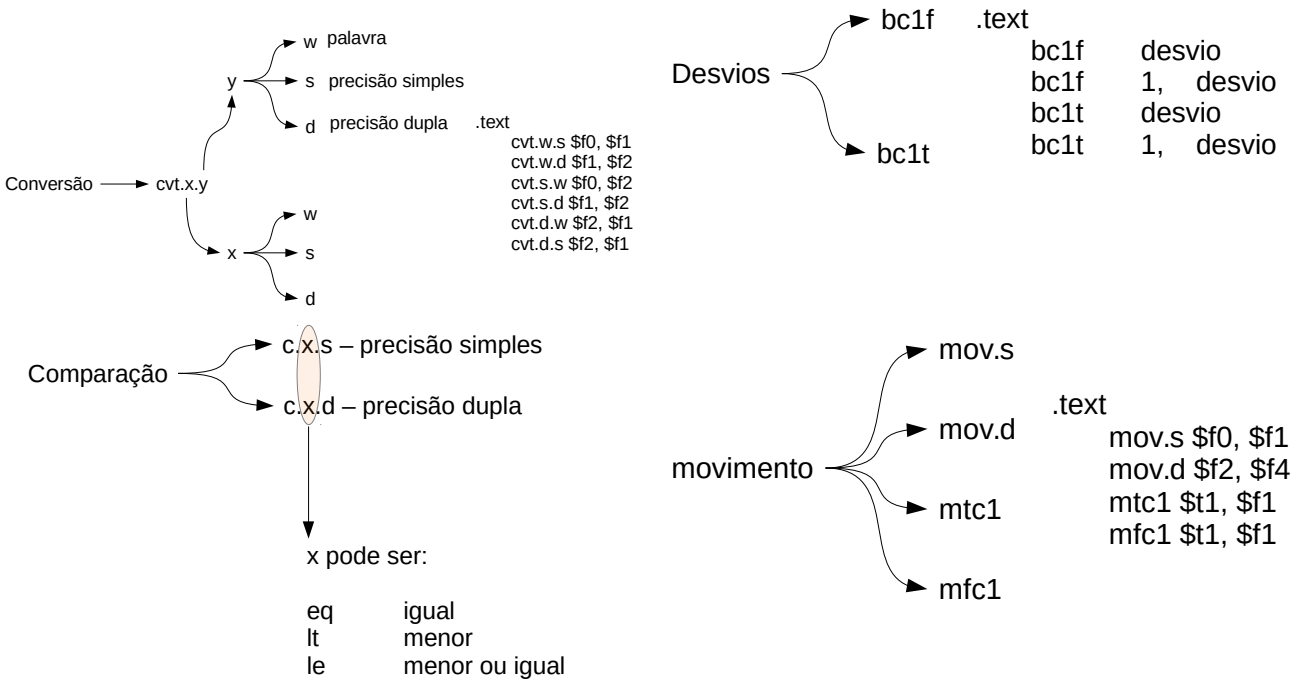

Tabela 7 – Instruções e valores dos sinais na unidade de controle do processador monociclo.

Controle	**Nome do Sinal**	**Formato R (0)**	**Lw (35)**	**Sw (43)**	**Beq (4)**
Entradas	OP5	0	1	1	0
	OP4	0	0	0	0
	OP3	0	0	1	0
	OP2	0	0	0	1
	OP1	0	1	1	0
	OP0	0	1	1	0
Saídas	RegDst	1	0	X	X
	UalFonte	0	1	1	0
	MemParaReg	0	1	X	X
	EscReg	1	1	0	0
	LerMem	0	1	0	0
	EscMem	0	0	1	0
	DvC	0	0	0	1
	UALOp1	1	0	0	0
	UALOp0	0	0	0	1

Tabela 8 – Operação da UAL para a combinação da UALOp e campo de função.

UALOp		Campo de Função						Operação da UAL	
UALOp1	UALOp1	F5	F4	F3	F2	F1	F0		
0	0	X	X	X	X	X	X	010	soma
X	1	X	X	X	X	X	X	110	subtração
1	X	X	X	0	0	0	0	010	soma
1	X	X	X	0	0	1	0	110	subtração
1	X	X	X	0	1	0	0	000	and
1	X	X	X	0	1	0	1	001	or
1	X	X	X	1	0	1	0	111	slt