



Nome: _____

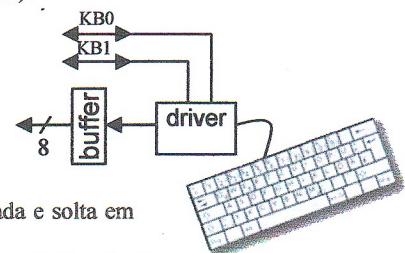
GABARITO

Matrícula: _____

Prova 2

1) (6.0) Desde o início da computação, o teclado é o principal dispositivo de entrada de dados. Um buffer é geralmente utilizado para armazenar os códigos das teclas pressionadas. Considerando um teclado “PS1” onde apenas uma tecla pode ser pressionada por vez. Ao se pressionar uma tecla o código ASCII correspondente é armazenado apenas uma vez em um buffer (registrador) de 1 byte, e a tecla ao ser largada o código 0x00 é armazenado no mesmo buffer. (Obs.: No laboratório foi utilizada interface MMIO).

1.1) (3.0) Interface por interrupção: A cada tecla pressionada, é setado um sinal interrupção KB0 para o processador. Esta interrupção ao ser processada coloca no registrador \$k0 do banco de registradores o valor presente no buffer, seta \$k1=1, reseta o sinal interrupção KB0 e continua com a execução normal do programa. Ao ser solta a tecla, é setado um sinal de interrupção KB1 para o processador. Esta interrupção ao ser processada reseta o registrador \$k1=0 (sem alterar o valor de \$k0), reseta o sinal interrupção KB1 e continua com a execução normal do programa. Considere que a detecção da interrupção é realizada somente no estado 0 do diagrama de estados do controle. Considere também que o teclado é um dispositivo lento, isto é, é impossível uma tecla ser pressionada e solta em menos de 10 períodos de clock.



- a) (2.0) Modifique adequadamente o Caminho de Dados (na folha em anexo) e desenhe a Máquina de Estados do Controle do processador multiciclo desenvolvido em aula a fim de implementar esse novo tipo de interfaceamento.
- b) (1.0) Escreva um procedimento em Assembly MIPS que faça o processador ler 2 dígitos em hexadecimal digitados no teclado (sem precisar teclar ENTER ao final) e retorne o seu valor em \$v0.

1.2) (3.0) Interface por instrução específica: Modifique o Caminho de Dados (na folha em anexo) e o Controle do MIPS uniciclo desenvolvido em aula de forma que a interface de acesso ao buffer de teclado seja feita através da incorporação de uma instrução específica que apenas lê o conteúdo do buffer do teclado e armazena em um registrador:

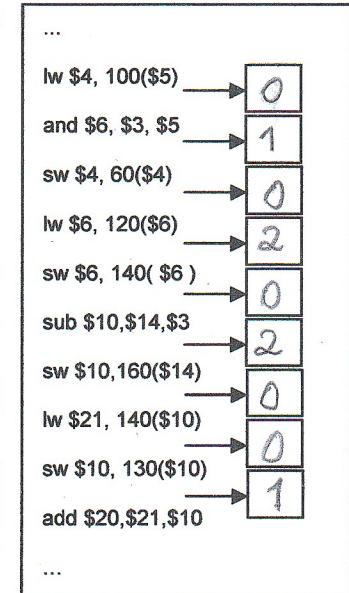
- a) (2.0) readkb \$rd # R[\$rd]={24'b0,buffer} sendo \$rt=0 e \$rs=0 OPCODE=0x00 FUNCT=0x2F
- b) (1.0) Escreva um procedimento em Assembly MIPS que faça o processador ler 2 dígitos em decimal digitados no teclado (pressionando ENTER ao final) e retorne o seu valor em \$v0.

2) (2.5) Considerando apenas os seguintes tempos de atraso das unidades operativas do caminho de dados de uma CPU MIPS:

Unidade	Tempo
Operação com a ULA	100ps
Leitura/Escrita no Banco de Registradores	50ps
Leitura da memória de Instruções ou Dados	200ps
Escrita na memória de Dados	300ps

Com relação ao trecho de programa em assembly MIPS ao lado:

- a) (0.5) Para a implementação uniciclo, qual será a maior frequência de clock utilizável? Qual o tempo de execução deste trecho de código neste caso?
- b) (0.5) Para a implementação multiciclo vista em aula, qual será a maior frequência de clock utilizável? Qual o tempo de execução deste trecho de código neste caso?
- c) (0.5) Qual o tempo de execução da implementação pipeline, se todos os hazards forem tratados apenas com inserção de bolhas? Indique no espaço reservado o número de bolhas necessário (obs.: Se não há necessidade de bolha coloque ‘zero’).
- d) (1.0) Qual o tempo de execução para implementação em pipeline, se os hazards forem tratados eficientemente pelo processador com forwarding e/ou inserção de bolhas? Preencha o pipeline esquemático na folha em anexo indicando as instruções, bolhas e forwards sugeridos. Considere que os registradores podem ser escritos e lidos no mesmo ciclo, que o branch é avaliado na etapa ID e previsto como não-tomado, e as instruções j e jal necessitam 2 ciclos.



3)(2.5) Suponha que tenhamos um processador com freqüência de clock de 4GHz e um CPI básico de 1,0, quando não há falhas de acesso na memória cachê de instruções de primeiro nível (L1). Considere que a memória RAM principal de 2 GiB tenha um tempo de acesso de 200ns. Implementando-se uma estrutura hierárquica de memórias cachê de 3 níveis (L1, L2 e L3) com as seguintes características:

Memória cachê	Tamanho	Taxa de Falhas Média	Tempo de Acesso
L1	2 KiB	10%	X
L2	64 KiB	2%	5ns
L3	2 MiB	0,5%	50ns

Considerando apenas o desempenho da cachê de instruções em operações tipo R e J, responda:

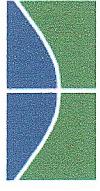
- a) (0.5) Quanto no máximo deve ser X?
- b) (0.5) Qual a CPI esperada do sistema sem o uso de memória cachê?
- c) (0.5) Quanto mais rápido será o sistema com apenas L1, em comparação com b).
- d) (0.5) Quanto mais rápido será o sistema com L1 e L2, em comparação com c)?
- e) (0.5) Quanto mais rápido será o sistema com L1, L2 e L3, em comparação com d)?

Boa Sorte!!!

Universidade de Brasília

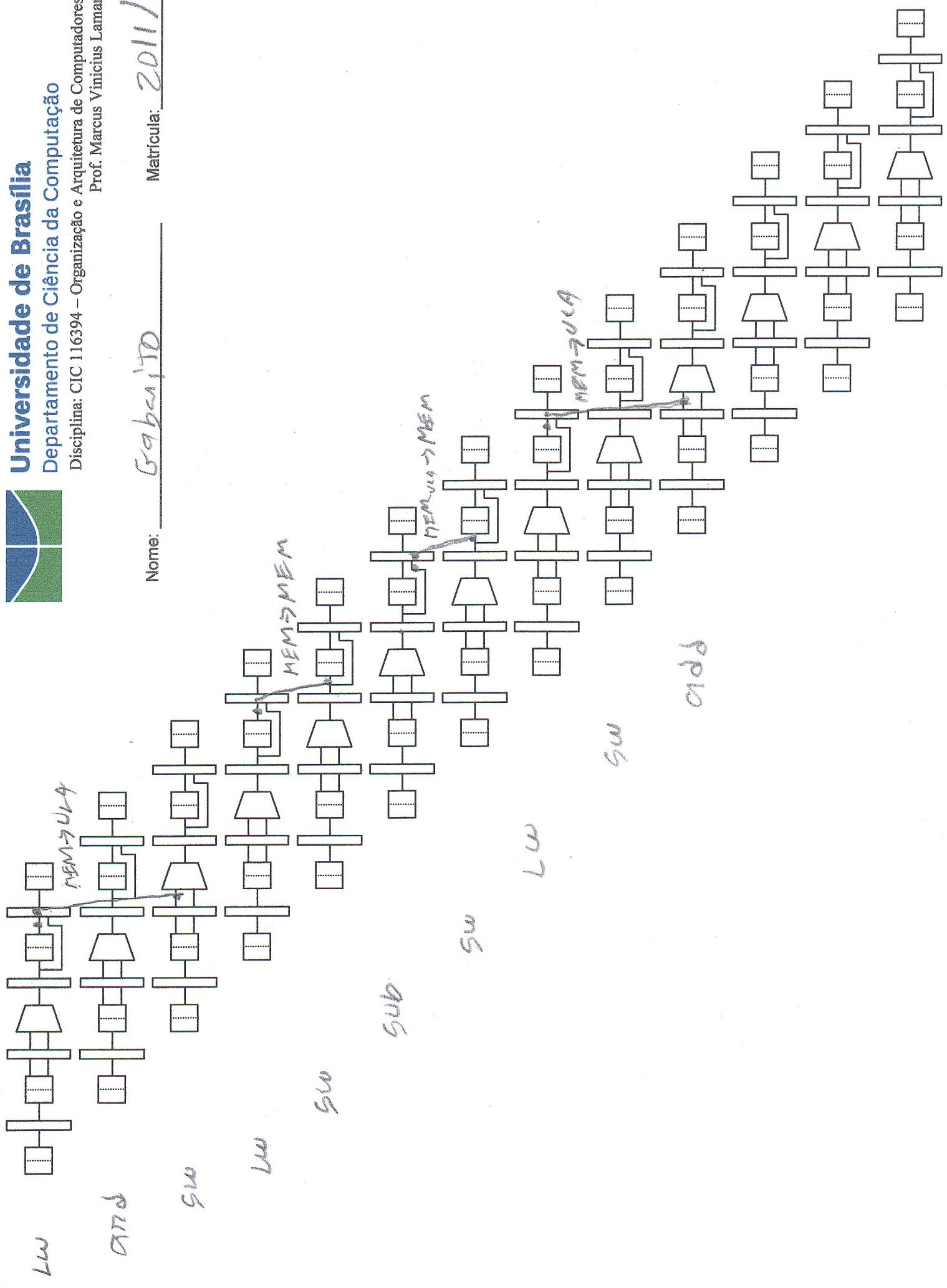
Departamento de Ciéncia da Computação

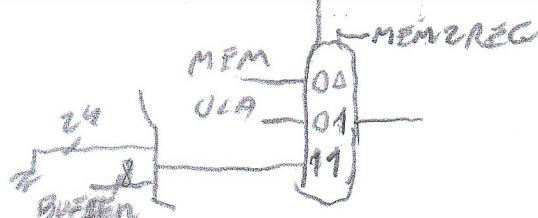
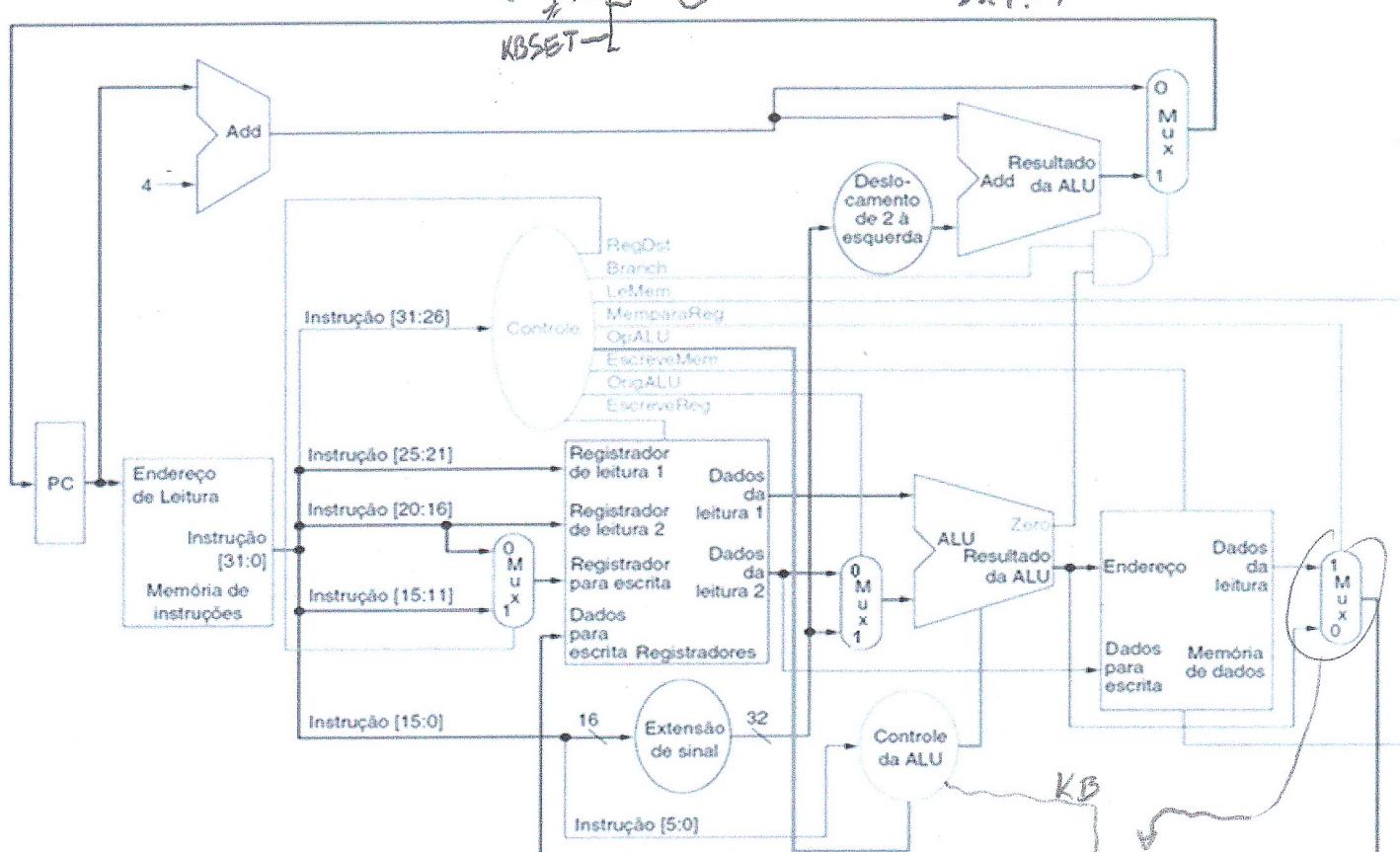
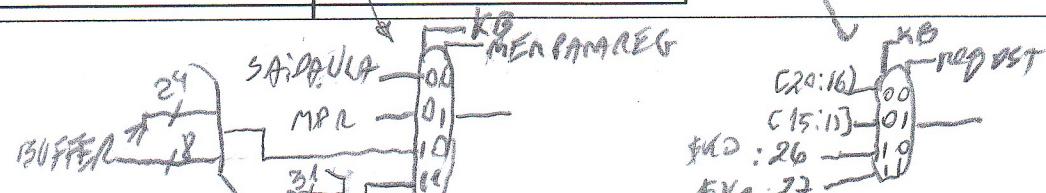
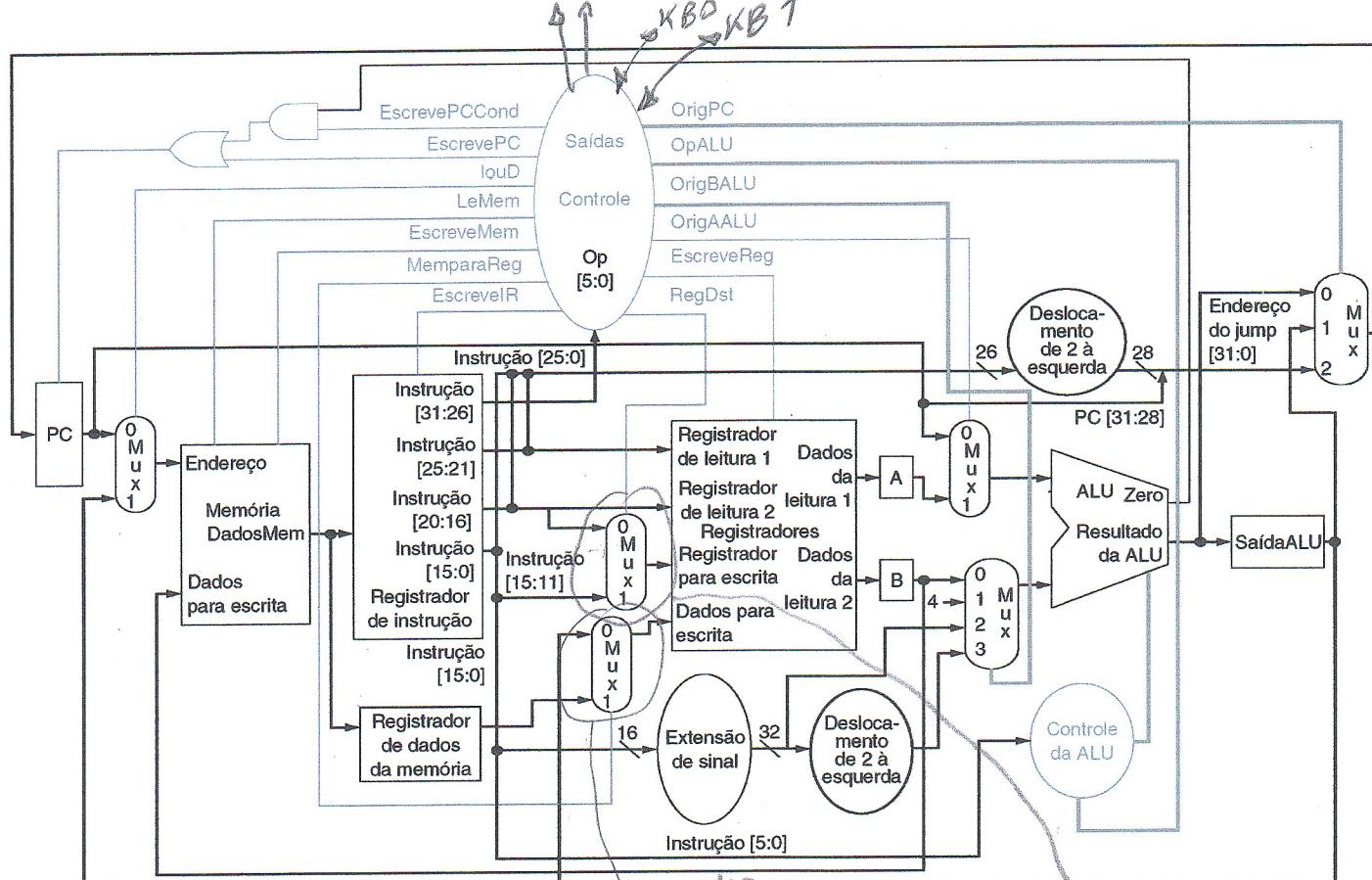
Disciplina: CIC 116394 – Organização e Arquitetura de Computadores
Prof. Marcus Vinicius Lamar



Matrícula: 2011102

Nome: Gabi TD



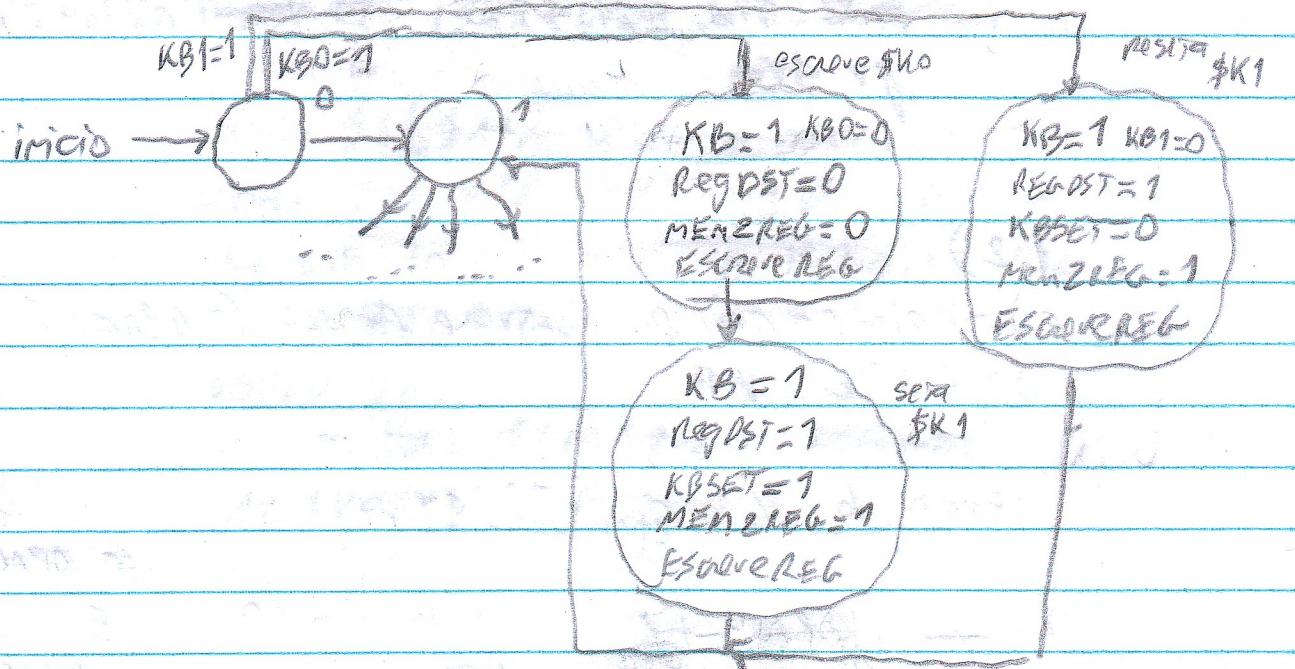


2º Prova

Gabarito

1.1)

a) interpretação de rotinas de estados



b) #supondo inicialmente nenhuma tecla pressionada

LE-HEX: Li \$t0, 1 # \$t0 = 1

LOOP1 : BEQ \$K1, \$ZERO, LOOP1 # Espera Pressionar

LOOP2 : BEQ \$K1, \$t1, LOOP2 # Espera soltar
MOVE \$t1, \$K0 # rsfia P1/\$t1

LOOP3 : BEQ \$K1, \$ZERO, LOOP3 # Espera Pressionar

LOOP4 : BEQ \$K1, \$t2, LOOP4 # Espera soltar
MOVE \$t2, \$K0 # rsfia P1/\$t2

SETA BIT 32

ORI \$t1, \$t1, 32 # isto é corrente

ORI \$t1, \$t1, 32 # A → a Mai→minúscula

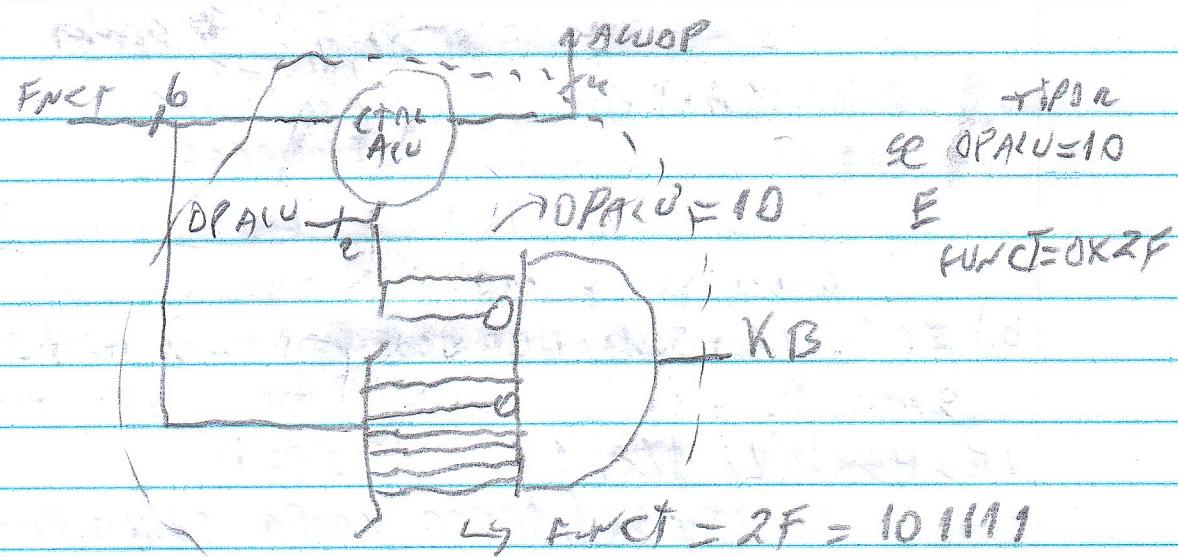
```

addi $t1,$t1,-48 # converte ASCII I->INT
addi $t2,$t2,-48 # " "
BLI $t1,10,PULAT # se FOR 47 na
addi $t1,$t1,-39 # A->10, B->11 etc.
PULAT: blt $t2,10,PULAT # igual a $t1/$t2
addi $t2,$t2,-39
PULAT: SLL $t1,$t1,4 # 1º dígito x 16
add $v0,$t1,$t2 # v0=1º dx 16 + 2º d
jr gra

```

10.2)

a) modificaçāo do controle da JUG P1 (graças à
do SINAL KB



b) LE-DEC: ReadKB \$t0 # Le 1º dígito
 BEQ \$t0,\$ZERO,LE-DEC # Espera Pressionar
 MOVE \$t1,\$t0
 LOOP1: ReadKB \$t0
 Bne \$t0,\$ZERO,LOOP1 # Espera Soltar

LE-2: readKB \$t0 # Leer teclado = 00000000
Beq \$t0, \$ZERO, LE-2 # Espera presión de tecla
move \$t2, \$t0

LOOP2: readKB \$t0
Bne \$t0, \$ZERO, LOOP2 # Espera soltar

ENTER: readKB \$t0 # Trae otra presión de tecla
Bne \$t0, 13, ENTER : ENTER

addi \$t1, \$t1, -48 # ASCII > int
addi \$t2, \$t2, -48 # sum verificación
li \$t0, 10 # 1º digito x 10
mult \$t1, \$t0
mflo \$t1
addi \$t0, \$t1, \$t2 # suma
jr \$ra

2) a) Unidicito, 1 regresa + letra

$$Cw: 200p + 50p + 100p + 200p + 50p = 600p$$

$$Sw: 200p + 50p + 100p + 300p = 650p \leftarrow$$

$$f_{max} = \frac{1}{650p} = 1,5384 \text{ GHz}$$

$$t_{uni} = 10 \times 1 \times 650p = 6,5ns \leftarrow$$

b) Multidicito, Etapa + Letra

$$f_{max} = \frac{1}{300p} = 3,333 \text{ GHz}$$

$$t_{mult} = (5+4+4+5+4+4+4+5+4+4) \times 330p \\ = 12,9ns \leftarrow$$

$$c) \text{t}_{PIPE} = (10+6) \times 300p = 4,8ns //$$

$$d) \text{t}_{PIPE} = 10 \times 300p = 3ns \\ 80\text{thas} \quad 4 \text{ forward}$$

3)

a) Se CPI=1 quando não há falhas de acesso à L1

$$\text{Logo } t_{L1} = \frac{1}{f} = \frac{1}{400} = 250ps = 0,25ns //$$

b) Sem cache:

$$\text{Penalidade} = \frac{2000}{0,25} = 800 \text{ ciclos}$$

$$\text{CPI total} = 800 // \quad \text{e mais 801} \rightarrow \text{ve clifabilidade}$$

$$= 2 \text{ ciclos!}$$

c) com L1 Penalidade = 800 ciclos

$$\text{CPI total} = 1 + 800 \times 0,1 = 81 \quad \eta = \frac{800}{81} = 9,87%$$

d) com L2

$$\text{Penalidade} = \frac{2000}{0,25} = 20 \text{ ciclos}$$

$$\text{CPI total} = 1 + 0,1 \times 20 + 0,02 \times 800 = 19$$

$$\eta = \frac{81}{19} = 4,26 //$$

e) com L3

$$\text{Penalidade} = \frac{2000}{0,25} = 200 \text{ ciclos}$$

$$\text{CPI total} = 1 + 0,1 \times 20 + 0,02 \times 200 + 0,005 \times 800 \\ = 11$$

$$\eta = \frac{19}{11} = 1,72 //$$