Universidade de Brasília
Departamento de Ciência da Computação

Disciplina: CIC 116394 – Organização e Arquitetura de Computadores – Turma A 2010/2
Prof. Marcus Vinicius Lamar

d_0 d_1 / d_2 d_3 d_4 d_5 d_6 d_7 d_8

Nome:_____ Matrícula: □□/□□□□□□□

Prova 1

(2.0) 1) Faça um procedimento em Assembly MIPS que possua a seguinte definição em C

void mult(float *C, float *A, float *B, int N)

e que realiza a operação de multiplicação matricial C=A×B entre as matrizes quadradas A, B de dimensões N×N.

Obs.: Considere a organização por linha da matriz na memória e A é um ponteiro para o elemento A(0,0).

(4.0) 2) Implemente um programa principal main: que:

(1.0) 2.1)Aloque estaticamente um espaço na memória de dados para as matrizes A,B e C de dimensões 3×3.

Obs.: Considere o endereço inicial do segmento de dados 0x10000000 (.data)

(2.0) 2.2)Leia os valores dos elementos das matrizes A e B do teclado, com a mensagem 'A(i,j)=', armazenando-os nos respectivos espaços alocados previamente.

(1.0) 2.3)Que chame de maneira adequada a função mult da questão 1).

(3.0) 3) A arquitetura x86 possui instruções bastante poderosas inexistente na arquitetura MIPS.

(1.0) 3.1) cpblk $t0, $s0, IMM #Copy Block – Instrução MIPS Tipo-I

Esta instrução copia um bloco de dados de tamanho IMM words, do endereço inicial da memória apontado pelo registrador $s0, para o endereço inicial da memória apontada pelo registrador $t0. Você como programador de um montador MIPS, dada esta pseudo-instrução, escreva o código real a ser usado em seu lugar no programa.

(1.0) 3.2) Escreva a equação que relaciona o tamanho do bloco IMM ao desempenho de um processador MIPS uniciclo de frequência de clock de 100MHz.

(1.0) 3.3) O que são as arquiteturas CISC e RISC?

(2.0) 4) Dado o código em Assembly MIPS ao lado, em que o label INICIO corresponde ao endereço 0x00400000 da memória.

O que será escrito na tela?

```
INICIO: lui $a0, 0x4d0d1d2
        la $t1,0x0810000B
        la $t2,JUMP
        sw $t1,36($t2)
JUMP:   beq $a0,$zero,FIM
        la $t1,0x00000000
        sw $t1,24($t2)
        li $a1, 0x00d3d4
FIM:    li $v0,2
        addi $a0,$a0,1
        mtc1 $a0,$f12
        syscall
```

BOA SORTE!

OAC - TURMA A 20102

1º PROVA

Gabarito

1)

```
MULT: li $t0,0
LOOPi: beq $t0, $a3, Fimi
       li $t1,0
LOOPj: beq $t1,$a3, Fimj
       mtc1 $ZERO, $f0
       li $t2,0
LOOPk: beq $t2, $a3, FimK
       mult $t0,$a3
       mflo $t5
       add $t5, $t5, $t2
       sll $t5, $t5, 2
       add $t5,$t5,$a2
       lwc1 $f1,0($t5)
       mult $t2,$a3
       mflo $t5
       add $t5, $t5, $t1
       sll $t5, $t5, 2
       add $t5, $t5, $a1
       lwc1 $f2,0($t5)
       mul.s $f2, $f2, $f1
       add.s $f0, $f0, $f2
       addi $t2, $t2, 1
       j LOOPk

FimK: mult $t0, $a3
      mflo $t5
      add $t5,$t5,$t1
      sll $t5,$t5,2
      add $t5,$t5,$a0
      swc1 $f0, 0($t5)

      addi $t1, $t1, 1
      j LOOPj

Fimj: addi $t0,$t0,1
      j LOOPi

Fimi: jr $ra
```

2) 2.1)

```
.data
A: .float 0,0,0,0,0,0,0,0
B: .float 0,0,0,0,0,0,0,0
C: .float 0,0,0,0,0,0,0,0
TA: .asciiz "\nAC"
TB: .asciiz "\nBC"
TC: .asciiz "\nC"
TV: .asciiz ","
TF: .asciiz ")="
    :
```

```
.text
main: la $a0, A
      la $a1, TA
      Li $a2, 3
      jal LER
      la $a0, B
      La $a1, TB
      Li $a2, 3
      jal LER
      La $a0, C
      La $a1, A
      La $a2, B
      Li $a3, 3
      jal MULT
      Li $v0, 10
      SYSCALL
```

```
LER: move $t3, $a0
     move $t4, $a1
     move $t2, $a2
     Li $t0, 0
LOOP_i: beq $t0, $t2, FIM_i
     Li $t1, 0
LOOP_j: beq $t1, $t2, FIM_j
     Li $v0, 4
     move $a0, $t4
     SYSCALL
     Li $v0, 1
     move $a0, $t0
     SYSCALL
     Li $v0, 4
     La $a0, TV
     SYSCALL
     Li $v0, 1
     move $a0, $t1
     SYSCALL
     Li $v0, 4
     La $a0, TF
     SYSCALL
     Li $v0, 6
     SYSCALL
     MULT $t0, $t2
     mflo $t5
     add $t5, $t1, $t5
     SLL $t5, $t5, 2
     add $t5, $t5, $t3
     swc1 $f0, 0($t5)
     addi $t1, $t1, 1
     j LOOP_j
FIM_j:
     add $t0, $t0, 1
     j LOOP_i
FIM_i:
     jr $ra
```

3) 3.1) CPBLK $t0, $s0, Imm

```
addi $sp, $sp, -12  # salva 2 registradores na pilha
sw $a0, 0($sp)
sw $a1, 4($sp)

addi $at, $zero, Imm
sll $at, $at, 2
loop: beq $at, $zero, Fim
add $a0, $at, $t0   # destino
add $a1, $at, $s0   # origem
lw $at, 0($a1)
sw $a1, 0($a0)
addi $at, $at, -4
j loop
Fim: lw $a1, 4($sp)
lw $a0, 0($sp)
addi $sp, $sp, 12
```

3.2) nº instruções = 5 + 1 + Imm x 7 + 3 (beq)

$N_i = 7 \times Imm + 9$

$t = N_i \times CPI \times T \qquad n = 1/t$

$$n = \frac{1}{(7 \times Imm + 9) \times 1 \times \frac{1}{100M}} =$$

$$n = \frac{100 \times 10^6}{7 \times Imm + 9} \;//$$

3.3) CISC: É uma arquitetura que possui um conjunto complexo de instruções

RISC: É uma arquitetura que possui um conjunto reduzido de instruções

4) $a0 = 0x4091

0x08100003 → 0000 1000 0001 0000 0000 0000 0000 0011

OPCode = 2

j 0000 0000 0100 0000 0000 0000 0001 1100

$f1 = j 0x0040002C

```
0040 0000  lui $a0, 0x4091
0040 0004  lui $at, 0x0810
0040 0008  ori $t1, 0x0003
0040 000C  lui $at, 0x0040
0040 0010  ori $at, $t2, 0x0018
0040 0014  sw $t1, 36($t2)  → JUMP +36 = 0040003C
Jump: 0040 0018  beq $a0, $zero, Fim
      001C  lui $at, 0x0000   Obs: La $t1, 0x00000000
      0020: ori $t1, $at, 0x0000  compilador esperto → addiu $t1, $zero, $zero
                                  convencional → lui + ori
      0024  sw $t1, 24($t2)  → Jump +24 = 00400038
      0028  addi $a0, $zero, 0x0023
Fim:  002C  addi $v0, $zero, 2
      0030  add.s $f12, $f0, 1
      0034  mtc1 $a0, $f12
      0038  syscall
      003C  j 0x00400002C
            Fim
```

SUBSTITUI POR NOP

Escreve repetidamente $a0 em float na tela

$a0 = 0x40910000 → $f12 = 4.53125

Logo:

4.53125 4.53125 4.53125 4.53125 ...