NOME: CÁSSIO CASTRO ARAUJO MATRÍCULA: 201420114

ORGANIZAÇÃO DE COMPUTADORES

Trabalho 3

1 – Traduza o seguinte trecho de programa em C para a linguagem de montagem do processador MIPS. Compile o programa em linguagem de montagem e simule no programa MARS (simule instrução a instrução de linguagem de máquina, passo a passo). O que faz este trecho de programa? Se necessário, mude o valor de i e refaça a simulação. Qual o valor de j após a execução do programa.

```
i = 10;
j = 0;
L1:
i = i - 1;
j = j + 1;
if (i == 0) goto L1
```

Durante a execução do trecho de código no programa MARS, vemos que ele só entrará no if final se o i iniciar com valor 1. Observamos também que ele entrará na label L1 mesmo que a mesma não seja chamada, tendo em vista que a verificação ocorre apenas no final de L1. **(Abrir mips1.asm na pasta para verificar o código)**

2 – Os vetores A, B e C estão armazenados no segmento de dados da memória. Estes são vetores de números inteiros. O vetor A tem 30 elementos, o vetor B possui 50 elementos e o vetor C contém 200 elementos. Traduza o seguinte trecho de um programa escrito em C, para a linguagem de montagem do MIPS. Compile e simule no MARS o seu programa em linguagem de montagem.

```
a[10] = b[23] + c[11];
b[45] = c[i] + 2;
if (i==j) goto L1;
c[j] = a[12] – 23;
goto L2;

L1:
a[k] = b[j] + c[i];

*L2:
[0] = a[k];
```

(Abrir mips2.asm na pasta para verificar o código)
***Obs: No caso de L2, foi perguntado em aula qual seria o vetor armazenado e foi informado que seria no vetor “a”. Acarretando que:**

```
L2:
a[0] = a[k]
```

3 – Escreva em linguagem de montagem um programa que realize o seguinte trecho de código:

```
if (a[i]==a[i+1]) goto L1 // se a[i] é igual a a[i+1] vá para L1
a[i] = x;
go to FIM // vá para FIM

L1: a[i] = y;

FIM:
```

(Abrir mips3.asm na pasta para verificar o código)

4 – Escreva em linguagem de montagem o seguinte comando if-else:

```
if (a[i]==x){ // se a[i] é igual a x, faça y igual a x
y = x;
}else{ // senão, faça y igual a a[i]
y = a[i];
}
```

(Abrir mips4.asm na pasta para verificar o código)

5 – Escreva em linguagem de montagem o seguinte trecho de código:

```
if (a[i] != b[i]){
      if(a[i] ==c[j]){
            b[i] = a[i] + c[j];
      }
}
```

(Abrir mips5.asm na pasta para verificar o código)

6 – Escreva em linguagem de montagem o seguinte laço while

```
while (i<j) {           // enquanto i<j faça a[i]=a[j]+x e i = i + k
a[i] = a[j] + x;
i = i + k;
}
```

(Abrir mips6.asm na pasta para verificar o código)

7 – Escreva um programa em linguagem de montagem para o seguinte trecho de programa, contendo o comando switch/case:

```
switch (k){
      case 10: a[i] = i + a[j]; break;
      case 11: a[i] = i – a[j]; break;
      case 12: switch(i){
                  case 0: a[i] = i;
                  case 1: a[i] = 2*i;
               }
               break;
}
```

(Abrir mips7.asm na pasta para verificar o código)

8 – Descreva como os procedimentos folha devem ser chamados. Quais os registradores que são usados para os argumentos da função? Quais são os registradores usados no retorno de valores do procedimento? Quais os registradores que devem ser preservados na chamada a um procedimento? Quais os registradores que não são preservados na chamada a procedimentos?

Os procedimentos folha devem ser chamados da seguinte forma:
jal procedimento, assim o mips vai direto para a instrução contida no procedimento informado e no final do procedimento folha temos a instrução jr $ra, que seria como uma função de retorno para a função chamadora.

Os registradores utilizados para os argumentos da função são:
$a0,$a1,$a2,$a3

Os registradores usados no retorno de valores do procedimento são:
$v0 e $v1

Os registradores que devem ser preservados na chamada a um procedimento são:
$s0,$s1,$s2,$s3,$s4,$s5,$s6,$s7

Os registradores que não são preservados na chamada a procedimentos são:
$t0,$t1,$t2,$t3,$t4,$t5,$t6,$t7,$t8,$t9

9 – Escreva um procedimento para ler um número inteiro. Se o número estiver fora da faixa de 0 a 10, o procedimento deve apresentar uma mensagem de erro.

(Abrir mips9.asm na pasta para verificar o código)

10 – Escreva um procedimento que calcula a soma de 4 variáveis (inteiros de 32 bits)

(Abrir mips10.asm na pasta para verificar o código)