Escola Politécnica da Universidade de Pernambuco

Curso: Engenharia de Computação

Disciplina : Arquitetura de Computadores

Turma : – 1EE (2ª Parte)

Data : 07/10/2016

1) Um programa tenta copiar duas palavras a partir do endereço armazenado no registrador Sa1 para o endereço do registrador Sa2, contando o número de palavras copiadas em Sv0. O programa pára de copiar quando encontra uma das palavras for maior do que 3 e menor do que 10, e a outra palavra for menor do que 4 ou maior ou igual a 9. As palavras finais devem ser copiadas, mas não devem ser contadas. (2,5 pontos)

2) O fragmento de código do mips a seguie processa um array e calcula valores importantes, armazenados nos registradores Sv0, Sv1, e Ss7. Suponha que o array possua 7000 palavras, indexadas de 0 a 6999, e que seu endereço-base $8000200C_{16}$ necessita ser armazenado em Sa0, e seu tamanho – 7000 já esteja armazenado em Sa1. Descreva em uma frase o que código faz, o que vai retornar em Sv0, Sv1, Ss7. (2,5 pontos)

```
Main:  Add  $a1, $a1, $a1
       Add  $a1, $a1, $a1
       Add  $v0, $zero, $zero
       Add  $t0, $zero, $zero
Outer: Add  $t4, $a0, $t0
       Lw   $t4, 0($t4)
       Add  $t5, $zero, $zero
       Add  $t1, $zero, $zero
Inner: Add  $t3, $a0, $t1
       Lw   $t3, 0($t3)
       Beq  $t3, $t4, Skip
       Addi $t5, $t5, 1
Skip:  Addi $t1, $t1, 4
       Bne  $t1, $a1, Inner
       Slt  $t2, $t5, $v0
       Bne  $t2, $zero, Next
       Add  $v0, $t5, $zero
       Add  $v1, $t4, $zero
Next:  Addi $t0, $t0, 4
       Bne $t0, $a1, Outer
       Add  $s7, $v0, $zero
       Addi $t0, $zero, 0
       Addi $t1, $zero, 1
Loop:  Slt  $t2, $s7, $t1
       Bne  $t2, $zero, Finish
       Add  $t0, $t0, $t1
       Addi $t1, $t1, 2
       J    Loop
Finish: Add $v0, $t0, $zero
```

Conta diferentes

Pega a maior diferença e o valor do elemento / ou último igual

$t1 = 1

$s7 = 3

t0 = 1
t1 = 3
t0 = 4
t1 = 5

$t1 = 1

$s7 = 8

t0 = 1
t1 = 3
t0 = 4
t1 = 5
t0 = 9
t1 = 7
t0 = 16
t1 = 9

5,0

LEE Arquitetura de Computadores 10/10/16

2ª Parte

Alexandre Diego Santos Silva

(1) base fonte -> $s1 base destino -> $s2

Programa: add $t0, $zero, $zero
add $v0, $zero, $zero

```
while : add $t1, $t0, $t0      # 2i
        add $t1, $t1, $t1      # 4i
        add $t2, $t1, $s1      # 4i + base (fonte)
        lw $t3, 0($t2)         # fonte[i]
        lw $t4, 4($t2)         # fonte[i+1]
        add $t1, $t1, $s2      # 4i + base (dest.no)
        sw $t3, 0($t1)
        sw $t4, 4($t1)
        addi $t1, $zero, 3     # $t1 = 3
        addi $t2, $zero, 10    # $t2 = 10
        slt $t9, $t3, $t2      # fonte[i] < 10
        beq $t9, $zero, verifica2 # e
        slt $t9, $t1, $t3      # fonte[i] > 3
        beq $t9, $zero, verifica2
        addi $t1, $t1, 1       # $t1 = 4
        addi $t2, $t2, -1      # $t2 = 9
        slt $t9, $t4, $t1      # fonte[i+1] < 4
        bne $t9, $zero, fim_while # ou
        slt $t9, $t4, $t2      # fonte[i+1] > 9
        beq $t9, $zero, fim_while
verifica2: slt $t9, $t3, $t1   # fonte[i] < 4
        bne $t9, $zero, verifica_and # ou
        slt $t9, $t3, $t2      # fonte[i] > 9
```

```
                bne $t9, $zero, continua
verifica_and:   addi $t1, $t1, -1        # $t1 = 3
                addi $t2, $t2, 1         # $t2 = 10
                slt $t9, $t4, $t2        # fonte[i+1] < 10
                beq $t9, $zero, continua # e
                slt $t9, $t1, $t4        # fonte[i+1] > 3
                bne $t9, $zero, fim_while
continua:       addi $v0, $v0, 1         # $v0 = $v0 + 1
                addi $t0, $t0, 2         # $t0 = $t0 + 2
                j while
fim_while:      JR $Ra
                                         # fim do programa
```

②

```
Main(){
int a1 = 7000;          //28000
int v0 = 0;

for(int t0 = 0; t0 < a1; t0++){
  t4 = array[t0];
  t5 = 0;
  for(int t1 = 0; t1 <= a1; t1++){
    t3 = array[t1];
    if(t3 != t4){
      t5++;
    }
  }
  if(t5 >= v0){
    v0 = t5;
    v1 = t4;
  }
}
```

```
s7 = v0;
t0 = 0;
t1 = 1;
while(s7 >= t1){
    t0 = t0 + t1;
    t1 = t1 + 2;
}
v0 = t0;
}
```

O código não faz o armazenamento do endereço base do array em $a0, logo não haverá execução de programa e caso haja será encontrado lixo de memória.

Para o caso do endereço ser armazenado em $a0 o código executará tendo sua execução baseada em:

- Faz uma contagem de elementos distintos a um elemento
- Armazena em $v1 o valor do elemento que menos se repete (ou possui uma maior contagem de elementos distintos)
- Armazena em $s7 o número de elementos distintos a $v1
- Armazena em $v0 a soma de todos os números ímpares positivos menores ou iguais ao número de elementos distintos, por exemplo:

se $s7 = 3 → $v0 = 1 + 3

se $s7 = 8 → $v0 = 1 + 3 + 5 + 7

se $s7 = 2000 → $v0 = 1 + 3 + 5 + 7 + ... + 1995 + 1997 + 1999

Ou seja:

$v1 ← Valor do elemento do array que menos se repete, caso haja mais de 1 com mesma quantidade, será o último a ocorrer

$s7 ← Número de elementos distintos a $v1.

$v0 ← Soma de ímpares positivos menores ou iguais a $s7