AC1 2013-2014

universidade de aveiro theoria poiesis praxis
Departamento de Electrónica, Telecomunicações e Informática

Arquitectura de Computadores I

Pedro Miguel Cabral

Aula 03

Estruturas de controlo de fluxo

1

AC1 2013-2014

A capacidade de decidir e realizar uma de várias tarefas com base num critério de verdade ou falsidade determinado durante a execução (conhecido como instrução *if()* nas linguagens de alto nível) é possibilitado no assembly do MIPS pelas instruções:

```
beq   Rsrc1, Rsrc2, Label      # branch if equal
bne   Rsrc1, Rsrc2, Label      # branch if not equal
```

e são conhecidas como "*branches*" (saltos) condicionais

2

AC1 2013-2014

beq Rsrc1, Rscr2, Label # branch if equal

Salta para a instrução situada no endereço representado por "Label", **se** os conteúdos dos registos **Rsrc1** e **Rsrc2** forem i**guais**

Como funciona?

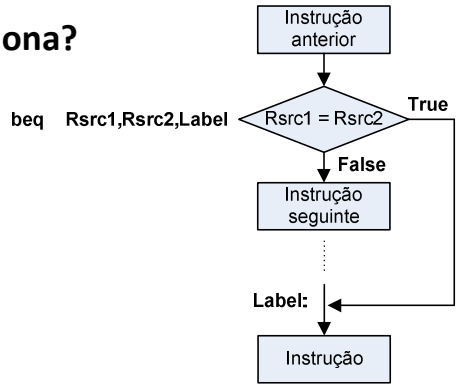
- Se a condição testada na instrução for verdadeira, o valor corrente do PC é substituído pelo endereço a que corresponde "Label"
- Assim, a próxima instrução a ser executada é a que se situa nesse endereço

3

AC1 2013-2014

beq Rsrc1, Rscr2, Label # branch if equal

Como funciona?

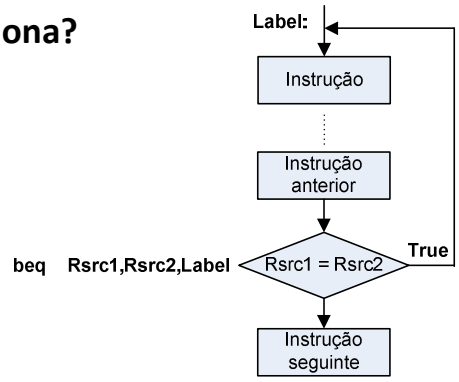


4

AC1 2013-2014

beq Rsrc1, Rscr2, Label # branch if equal

Como funciona?



5

AC1 2013-2014

São ainda suportadas um conjunto de instruções que comparam directamente com zero:

bltz Rsrc, Label # Branch if Rsrc < 0

blez Rsrc, Label # Branch if Rsrc ≤ 0

bgtz Rsrc, Label # Branch if Rsrc > 0

bgez Rsrc, Label # Branch if Rsrc ≥ 0

6

AC1 2013-2014

Consideremos os seguintes trechos de código:

```
if (i >= j) {
    xxxxxx;
} else {
    yyyyyy;
}...
```

```
for (n = 0; n < 100; n++)
{
    xxxxxx;
}
...
```

```
n = 0;
do
{
    xxxxxx;
    n++;
}while (n < 100);
...
```

```
n = 0;
while (n < 100)
{
    xxxxxx;
    n++;
}
...
```

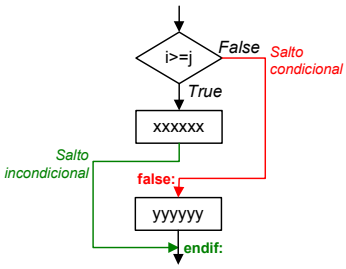
7

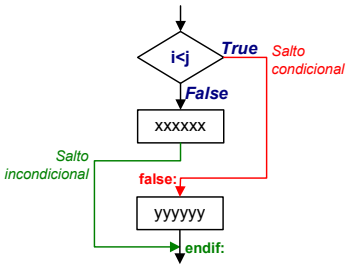
AC1 2013-2014

1º Exemplo:

```
if (i >= j) {
    xxxxxx;
} else {
    yyyyyy;
}...
```

Necessário adaptar o salto condicional para que este salte quando a condição for verdadeira (tal como nos *branches*).






```
$t0  ↔  i
$t1  ↔  j
        blt     $t0, $t1, false  # if (a >= n) {
        xxxxxx                   #     xxxxxx;
        j       endif            # }
false:                           # else {
        yyyyyy                   #     yyyyyy;
endif:  ...                      # }
```

j significa ***jump*** e representa um salto **incondicional** para o endereço indicado

8

AC1 2013-2014

2º Exemplo:

```
for (n = 0; n < 100; n++)
{
    xxxxxx;
}
...
```

```
n = 0;
while (n < 100)
{
    xxxxxx;
    n++;
}
...
```

Estes dois exemplos são funcionalmente equivalentes!

Operações a executar antes e fora do corpo do ciclo (inicializações)

Condição de continuação da execução do ciclo

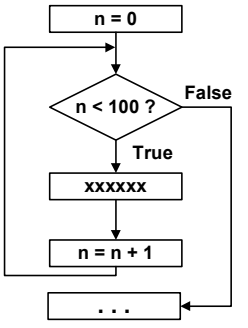
Operações a realizar no fim (dentro) do corpo do ciclo

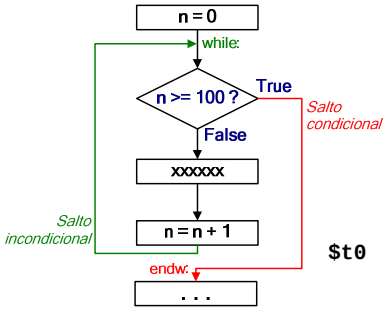
AC1 2013-2014

2º Exemplo:

```
for (n = 0; n < 100; n++)
{
    xxxxxx;
}
...
```

```
n = 0;
while (n < 100)
{
    xxxxxx;
    n++;
}
...
```





O salto condicional necessita de ser modificado de forma a saltar quando a condição for verdadeira.

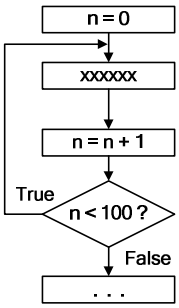
```
$t0  ↔  n

        li    $t0, 0
while:
        bge   $t0, 100, endw
        xxxxxx
        addi  $t0, $t0, 1
        j     while
endw:
```

AC1 2013-2014

3º Exemplo:

Ao contrário do **for()** e do **while()**, o corpo do ciclo **do...while()** é executado incondicionalmente pelo menos uma vez!

```
n = 0;
do
{
    xxxxxx;
    n++;
}while (n < 100);
...
```




```
$t0 ↔ n

        li      $t0,0           # n = 0;
do:                             # do {
        xxxxxx                  #     xxxxxx;
        addi    $t0, $t0, 1     #     n = n + 1;
        blt     $t0, 100, do    # } while(n < 100);
        ...
```

Desta feita, o teste condicional é efectuado no fim do ciclo

11

AC1 2013-2014

Resumindo:

- As estruturas do tipo **ciclo** incluem, geralmente, uma ou mais instruções de inicialização de variáveis, executadas antes e fora do mesmo
- No caso do **for()** e do **while()** o teste condicional é executado no início do ciclo
- No caso do **do...while()** o teste condicional é efectuado no fim do ciclo
- Na tradução de um **for()** para *Assembly*, o terceiro campo é incluído no fim do corpo do ciclo.

12