

MC404AE - Organização Básica de Computadores e Ling. Montagem

Controlando Fluxo de Execução

Prof. Allan M. de Souza

Agenda

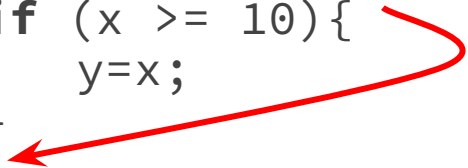
- Sentenças condicionais
- Laços de repetição
- Chamadas e retorno de funções
- Exemplos

Controle do fluxo de execução

Sentença condicional “**Se-Então**” (if-then)

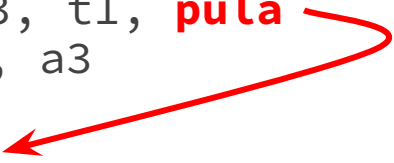
C/C++

```
...  
if (x >= 10) {  
    y=x;  
}  
...
```



Assembly

```
# x está em a3  
# y está em a4  
...  
li    t1, 10  
blt   a3, t1, pula  
mv    a4, a3  
  
pula:  
...
```



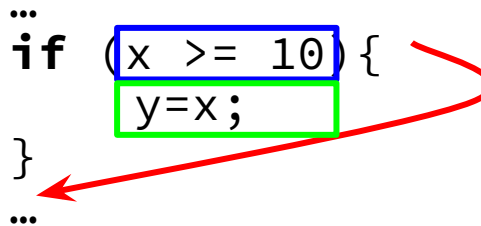
blt rs1, rs2, alvo: salta para o alvo se o valor em rs1 for menor que o valor em rs2 (supondo números com sinal). Para números sem sinal use “bltu”.

Controle do fluxo de execução

Sentença condicional “**Se-Então**” (if-then)

C/C++

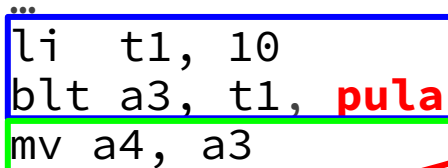
```
...  
if (x >= 10) {  
    y=x;  
}  
...
```



Assembly

```
# x está em a3  
# y está em a4
```

```
...  
li t1, 10  
blt a3, t1, pula  
mv a4, a3
```



pula:

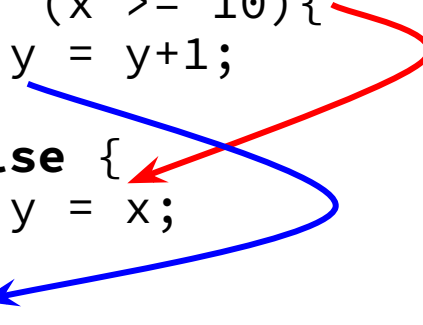
```
...
```

Controle do fluxo de execução

Sentença condicional “**Se-Então-Senão**” (if-then-else)

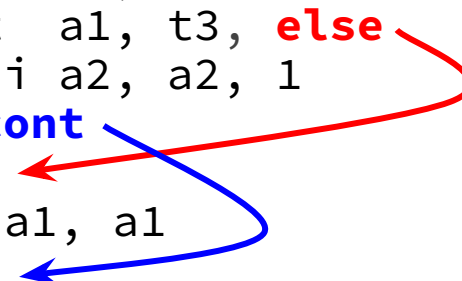
C/C++

```
...  
if (x >= 10) {  
    y = y+1;  
}  
else {  
    y = x;  
}  
...
```



Assembly

```
# x está em a1  
# y está em a2  
...  
li    t3, 10  
blt   a1, t3, else  
addi  a2, a2, 1  
j     cont  
else:  
mv    a1, a1  
cont:  
...
```



Controle do fluxo de execução

Sentença condicional “**Se-Então-Senão**” (if-then-else)

C/C++

```
...
if (x >= 10) {
    y = y+1;
}
else {
    y = x;
}
...
```

Assembly

```
...
# x está em a1
# y está em a2
...
li    t3, 10
blt   a1, t3, else
addi  a2, a2, 1
j     cont
else:
    mv  a2, a1
cont:
...
```

Controle do fluxo de execução

Sentença condicional com múltiplas condições

C/C++

```
...
if ((x>=10) && (y<20)){
    x = y;
}
...
```

Assembly

```
# x está em a1
# y está em a2
...
li t1, 10
blt a1, t1, pula
li t1, 20
bge a2, t1, pula
mv a1, a2
```

pula:

...

Laços de Repetição

Laços de Repetição

Laço de repetição “enquanto” (while)

C/C++

```
...
i = 0;
while (i < 20){
    y = y+3;
    i = i+1;
}
...
```

Assembly

```
# x está em a1
# y está em a2
...
li a1, 0
enquanto:
    li t1, 20
    beg a1, t1, cont
    addi a2, a2, 3
    addi a1, a1, 1
    j enquanto
cont:
...
```

Laços de Repetição

Laço de repetição “enquanto” (while)

C/C++

```
...
i = 0;
while (i < 20) {
    y = y+3;
    i = i+1;
}
...
```

Assembly

```
# x está em a1
# y está em a2
```

```
...
li a1, 0
enquanto:
    li t1, 20
    beg a1, t1, cont
    addi a2, a2, 3
    addi a1, a1, 1
] enquanto
cont:
...
```

Laços de Repetição

Laço de repetição “ para ” (for)

Assembly

C/C++

```
...
for (i=0; i<10; i++){
    y = y+2;
}
...
```

```
# x está em a1
# y está em a2
```

```
...
li a1, 0
for:
li t1, 10
bge a1, t1, cont
addi a2, a2, 2
addi a1, a1, 1
j for
cont:
```

Laços de Repetição

Laço de repetição “faça-enquanto” (do-while)

C/C++

```
...  
i=0;  
do {  
    y = y+2;  
    i = i+1;  
} while (i < 10);  
...
```

Assembly

```
# x está em a1  
# y está em a2  
...
```

Como podemos implementar?

Laços de Repetição

Laço de repetição “faça-enquanto” (do-while)

C/C++

```
...  
i=0;  
do {  
    y = y+2;  
    i = i+1;  
} while (i < 10);  
...
```

Assembly

```
# x está em a1  
# y está em a2  
...  
li a1, 0  
li t1, 10  
  
do:  
    addi a2, a2, 2  
    addi a1, a1, 1  
    blt  a1, t1, do  
...  

```

Chamada e retorno de funções

Chamando Funções

C/C++

```
...  
i = soma(2,3)  
...
```

Assembly

```
# i está em a3  
# Soma:  
# parâmetros em a0 e a1  
# retorno em a0  
...  
li a0, 2  
li a1, 3  
jal soma  
mv a3, a0  
...
```

Chamando Funções

C/C++

```
int soma(int a, int b){  
    return a+b;  
}
```

Assembly

```
# i está em a3  
# Soma:  
# parâmetros em a0 e a1  
# retorno em a0  
...  
soma:  
    add a0, a0, a1  
    ret  
...
```

Pseudo-instrução ret é convertida pelo montador para: **jalr x0, ra, 0**

Salta para o endereço **ra+0** e grava **PC+4** no registrador **x0** (ou seja, **descarta** PC+4)

Exemplos

Exemplo 1

Traduza o seguinte programa para ling. de montagem RV32

```
/* Global array */  
int numbers[10];  
  
/* Returns the largest value from array numbers. */  
int get_largest_number() {  
    int largest = numbers[0];  
    for (int i=1; i<10; i++) {  
        if (numbers[i] > largest)  
            largest = numbers[i];  
    }  
    return largest;  
}
```

Exemplo 2

Escreva uma função chamada **busca_caractere** que verifica se uma cadeia de caracteres terminada em zero possui um determinado caractere.

Entrada:

- a0: endereço inicial da cadeia
- a1: caractere a ser procurado

Retorna (em a0):

- endereço da primeira posição da cadeia onde a letra ocorre; ou
- o valor zero, caso não seja encontrado.

Exemplo 3

- Escreva um trecho de programa que determina qual o maior valor de um vetor de números inteiros de 32 bits sem sinal cujo endereço inicial é dado em a2.
- Inicialmente, a3 contém o número de valores presentes na cadeia; suponha que o valor em a3 é > 0 .
- Ao final do trecho, a0 deve conter o valor máximo e a1 deve conter o endereço de memória onde se encontra o valor máximo.

Exemplo 1 - Resp

```

.data
numbers: .skip 40      # int numbers[10];
.text
get_largest_number:
    la t1, numbers      # t1 <= &numbers[0]
    lw a0, (t1)         # largest <= numbers[0]
    li t2, 0            # i <= 1
    li t3, 10
    li t5, 4

for:
    bge t2, t3, cont    # Sai do laço se i >= 10
    mul t4, t2, t5      # t4 <= i*4
    add t4, t4, t1      # t4 <= &numbers[0] + i*4
    lw t4, (t4)         # t4 <= numbers[i]
    ble t4, a0, end_if  # Se t4 <= largest salta
    mv a0, t4           # senão: largest <= t4

end_if:
    addi t2, t2, 1      # i <= i+1
    j for

cont:
    ret

```

Exemplo 2 - Resp

busca_caractere:

laco:

```
lbu t1, 0(a0)           # Carrega o caractere atual
beq t1, zero, nao_enc   # Caso seja zero, retorna nao encontrou
beq t1, a1, encontrou   # Caso seja igual ao caractere de interesse
                        # retorne o endereço em a0
addi a0, a0, 1          # Caso contrário, avança com o apontador a0
j laco
```

nao_enc:

```
li a0, 0
```

encontrou:

```
ret
```

Exemplo 3 - Resp

lw a0, (a2)
mv a1, a2
addi a3, a3, -1

laco:

ble a3, zero, **terminei**
addi a2, a2, 4
lw t1, (a2)
bleu t1, a0, **pula_valor**
mv a0, t1
mv a1, a2

pula_valor:

addi a3, a3, -1
j laco

terminei: