Assembly RISC-V

Como Programar Nisso?

Download RARS
Exercício
Dúvidas

github.com/thaleslim/aulas

Saltos de Execução

Condicionais

→ **Qual o comportamento/essência de uma estrutura condicional?**

Quando o programa se depara com uma estrutura condicional, avalia a condição e executa, ou não, as instruções contidas nela.

Em outras palavras, caso a condição seja falsa o programa "pula" o conjunto de instruções, portanto, existem 2 "ramos" ou possibilidades de execução: executar ou "pular".

***Branch* → Ramo**

Instruções

Compõem a lógica do programa

→ Condicionais:

✔ Branch if EQual → beq t0, t1, label

⇒ Se t0 == t1, "pula" para endereço <label>

✔ Jump → j label *

⇒ "Pula" para endereço <label>, incondicionalmente

* Pseudoinstrução

Condicionais

```
int main(){

        int a = 0;
        scanf(“%d”, &a);
        if ( a == 0 )
                printf(“Zero!!!!”);
        else
                printf(“Não Zero!!!”);

        return 0;
}
```


Condicionais

E se trocássemos:
j ~~fora~~ if

```
.data
eh_zero: .asciz "É zero!!!"
not_zero:.asciz "Não é zero!!!"
.text
        li a7, 5                    # código serviço: Ler inteiro
        ecall                       # chamada de sistema
        li a7, 4                    # código serviço: Imprimir string
if:     bne a0, zero, else          # if ( a0 != 0 ) “pula” para else, senão...
              la a0, eh_zero        #     { a0 = &“É zero!!!” }
        j fora                      # fim do if → salto incondicional
else:la a0, not_zero                # else{ a0 = &“Não é zero!!!” }
fora:ecall                          # chamada de sistema
        li a7, 10                   # código serviço: Encerra programa
        ecall                       # chamada de sistema
```

While

→ Qual o comportamento/essência de uma estrutura de repetição?

Podemos imaginar como uma estrutura condicional e um comando que "pula para trás", voltando para a condição.

```
if ( a > 0 ) {
        printf("%d...", a--);
        repete if;
}
```

While

```
int main(){

        int a = 0;
        scanf(“%d”, &a);

        while ( a > 0 )
                printf(“%d...”, a--);

        printf(“Kabum!!!\n”);
        return 0;
}
```

While

if + "repete" → while

```
[...]
        li a7, 4                    # código serviço: Imprimir string
        mv s0, a0                   # s0 = a0
while:  bltz s0, fora               # if ( s0 < 0 ) "pula" para fora, senão...
                mv a0, s0           #     a0 = s0
                li a7, 1            #     código serviço: Imprimir int
                ecall               #     chamada de sistema
                addi s0, s0, -1     #     s0--
                la a0, dotdotdot    #     a0 = &"..."
                li a7, 4            #     código serviço: Imprimir string
                ecall               #     chamada de sistema
        j while                     # repete → volta ao inicio do loop
fora:   [...]
```

Instruções

Compõem a lógica do programa

→ Saltos:

- ✔ Jump And Link → jal t0, label

  ⇒ Salva o endereço de retorno* em t0 e “pula” para <label>

- ✔ Jump And Link Register → jalr t0, Imm(t1)

  ⇒ Semelhante ao anterior, mas “pula” para <t1+Imm>

- ✔ Jump → j label → jal zero, label

“pula”, mas sabe voltar

Funções

Porque ponteiros?

```
void zerar ( int * a ){
        if ( *a != 0 )
                *a = 0;
}

int main(){

        int a = 0;
        zerar(&a);
        return 0;
}
```

Funções

```
.text
      li a0, 0          # 1ª a0 = 0
      jal ra, zerar     # 2ª salva o endereço da próxima instrução e
                        #     "pula" para o endereço zerar
      li a7, 10         # 5ª código encerra programa
      ecall             # 6ª system call

## Função zerar
zerar:  beq a0, zero, back  # 3ª if (a0 == 0) "pula" pra back
      li a0, 0          # if (a0 != 0) {a0 = 0;}
back:
      ret               # 4ª jalr zero, 0 (ra) → retorna da função
```

Exercícios

Exercícios de Fixação

1) Crie uma função que calcule a média entre dois números

Input:
2 inteiros
Output:
1 único número inteiro

Exemplo:

INPUT	OUTPUT
2 3	2

Exercícios de Fixação

2) Crie uma função que calcule o fatorial de um número

Input:
1 inteiro
Output:
1 único número inteiro

Exemplo:

INPUT	OUTPUT
2	2
3	

Referências Bibliográficas

- RARS: RISC-V Assembler Runtime Simulator

Buscar por *Latest Release*, clicar em *Assets* e <rars>.jar

- Guia Prático RISC-V: Atlas de uma arquitetura aberta

The End

Thales Menezes