Procedimentos (MIPS)

Marcos Monteiro Junior

Procedimentos

- O desenho de um programa pode ser simplificado subdividindo o problema a resolver em um ou mais sub-problemas, seguindo a abordagem de dividir para conquistar.

- Estas subrotinas permitem escrever um programa complexo de uma forma modular, permitindo assim uma melhor depuração de erros e também a reutilização de sub-rotinas.

- Quando um procedimento (o *caller*) chama outro procedimento (o *callee*), o controle do programa é transferido do *caller* para o *callee.*

- As instruções que tornam possível o correto funcionamento deste mecanismo são as instruções jal e jr.

Os quatro passos necessários para executar um procedimento são:

1. Guardar o endereço de retorno. O endereço de retorno é o endereço da instrução que surge imediatamente após o ponto da chamada.

2. Chamar o procedimento.

3. Executar o procedimento.

4. Recuperar o endereço de retorno e retornar ao ponto inicial (que é o ponto da chamada).

A arquitetura do MIPS fornece duas instruções que são utilizadas em conjunção para realizar a chamada e retornar:

- jal nome_do_procedimento
- jr $ra

Importante

- É obrigatório salva/guardar o endereço de retorno ($ra) quando se entra num procedimento que chama outros procedimentos.

Registradores

- O software do MIPS separa 18 dos registradores em dois grupos (**convenção**)
  - $t0-$t9: 10 registradores temporários que não são preservados pelo procedimento chamado
  - $s0-$s7: 8 registradores que precisam ser preservados em uma chamada (se forem usados, o procedimento chamado os salva e restaura)

Pilha

- Registrador $sp
  - Responsável por apontar o topo da pilha
  - Também por convenção, deve-se salvar os registradores utilizados no procedimento “chamador”.
- Registradores utilizados para parâmetros
  - $a0->$a3
- Registradores utilizados para retorno
  - $v0,$v1

Ajustar o valor da pilha

- Addi $sp, $sp, -4
    - Ajuste para uma word
- Sw reg, 0($sp) #push
- Addi $sp,$sp, 4 #pop

Passo a passo

- **Chamando um Procedimento**

O *caller* (procedimento que chama, ou procedimento "chamante") realiza o seguinte antes de chamar outro procedimento:

- 1. Guarda quaisquer registos que não são preservados ao longo de chamadas (i.é, $a0-$a3 e $t0-$t9) e que se espera que sejam usados após a realização da chamada.
- 2. Passa os parâmetros: os 4 primeiros são passados em $a0-$a3, os restantes são passados pela pilha.
- 3. Salta para o procedimento usando a instrução jal.
- 4. Após a chamada, ajusta o $sp para apontar acima dos argumentos passados na pilha (isto é equivalente a fazer um *pop*, só que nada é lido da pilha).

O *callee* (procedimento chamado) faz o seguinte:

- 1. Guarda o $ra se o procedimento for chamar outro procedimento.
- 2. Guarda quaisquer registos $s0-$s7 que o procedimento modifique.
- 3. Realiza o que tem que realizar.
- 4. No caso de retornar um valor, coloca esse valor no registo $v0.
- 5. Restaura todos os registos previamente guardados (*pop* da pilha).
- 6. Retorna, executando jr $ra.

Referências

- Apostila
  - Arquitectura de Computadores – *Guia dos Laboratórios* Pedro F. Campos