



Suporte a Subrotinas

Universidade Federal de Uberlândia
Faculdade de Computação
Prof. Dr. rer. nat. Daniel D. Abdala

Na Aula Anterior ...

- Introdução;
- Saída de Dados;
- Entrada de Dados;
- Outros Serviços do Sistema;
- Término do programa;

Nesta Aula

- Revisão acerca das instruções de salto;
- Passos para a execução de um procedimento;
- Registradores para suporte a sub-rotinas;
- Pilha para argumentos e dados;
- Estrutura geral de um procedimento;
- Procedimentos aninhados.

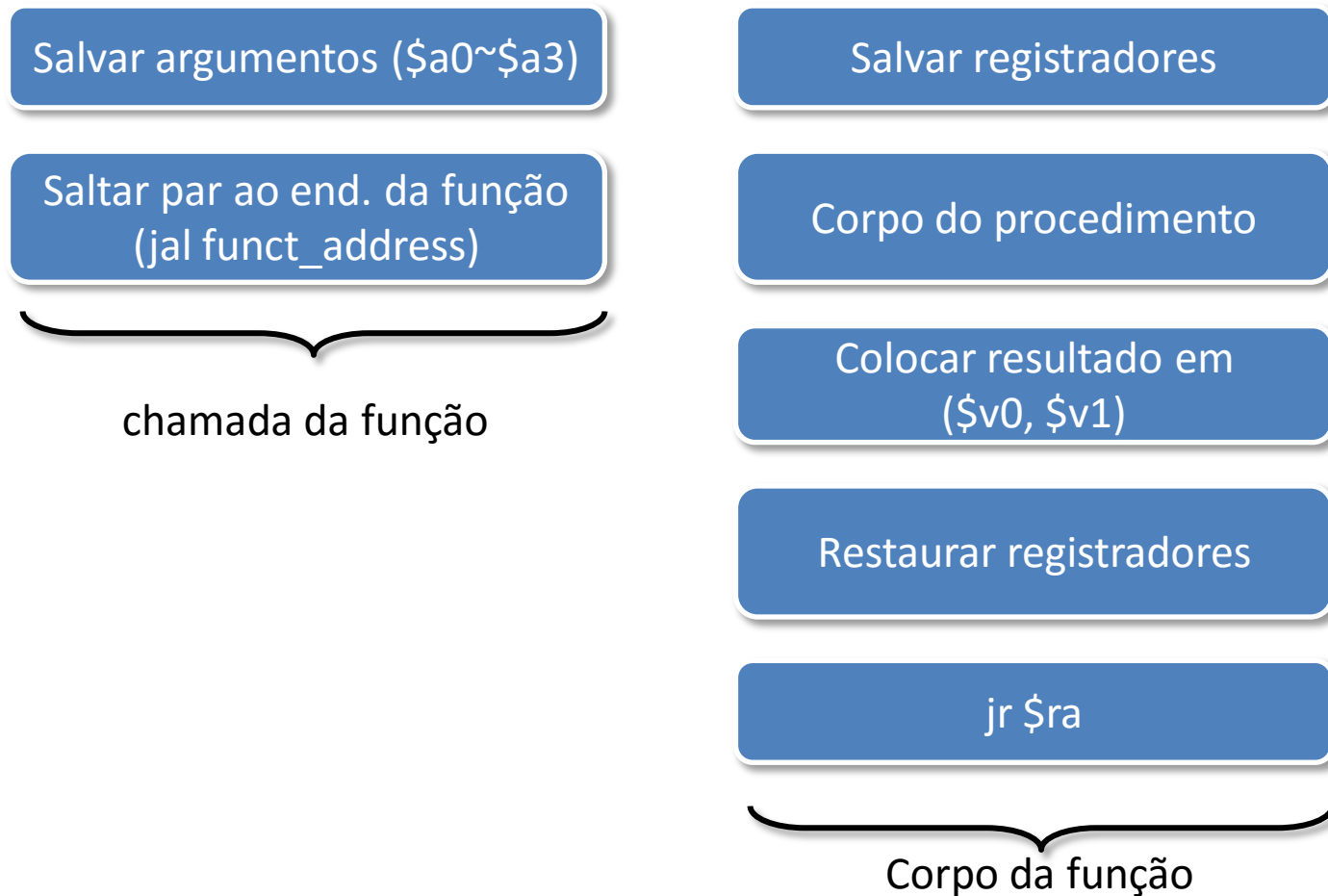
Revisão – Instruções de Salto

- `jr` → jump register
 - Usado para saltos absolutos;
 - Usado em sub-rotinas (retorno da sub-rotina);
 - Endereços de até 32 bits (capacidade do registrador);
 - Ex: `jr $s0`
- `jal` → jump and link
 - Usado em sub-rotinas;
 - Seta `$ra` para o endereço de PC+4 (prox. instrução);
 - Salta para o endereço especificado;
 - Ex: `jal LABEL`
- `jalr` → jump and link register
 - Usado em sub-rotinas;
 - Seta `$ra` para PC+4 e salta para a pos. de mem. em `$s0`;
 - Ex: `jalr $s0`

Passos para a Execução de um Procedimento

1. Colocar os parâmetros em algum lugar acessível pelo procedimento;
2. Transferir o controle para o procedimento;
3. Adquirir os recursos armazenados necessários para a execução do procedimento;
4. Executar o código do procedimento;
5. Colocar os valores de retorno em algum lugar acessível ao programa chamador;
6. Retornar o controle de execução ao ponto de origem.

Estrutura Geral de um Procedimento



Registradores para Suporte a Sub-rotinas

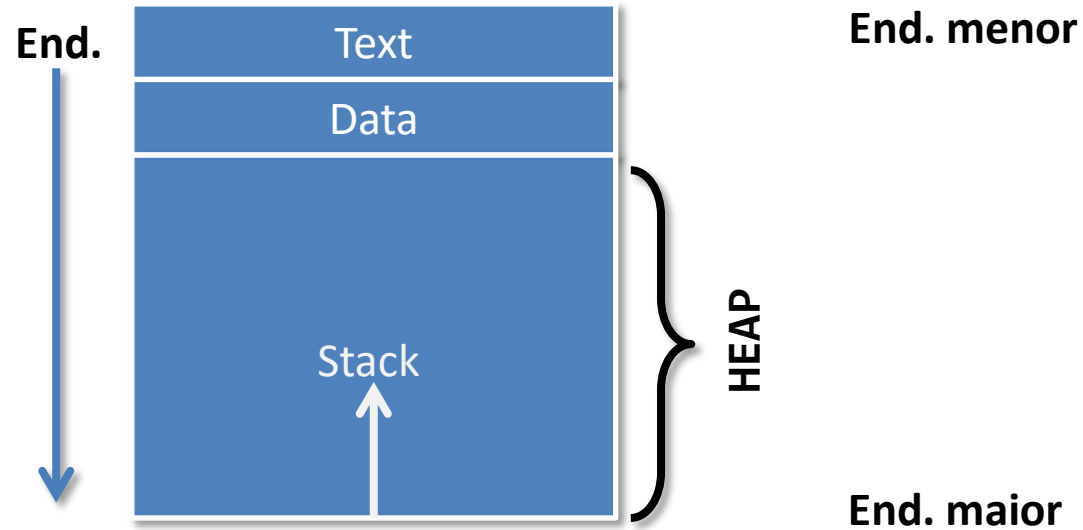
- \$a0~\$a3 – registradores para passagem de argumentos;
- \$v0,\$v1 – registradores para retorno de resultados;
- \$ra – registrador para endereço de retorno da sub-rotina.

Exemplo

```
int leaf(int g, int h, int i, int j){  
    int f;  
    f =(g+h)-(i+j);  
    return f;  
}
```

leaf:	addi	\$sp,\$sp,-12	}	overhead de suporte a funções
	sw	\$t1,8(\$sp)		
	sw	\$t0,4(\$sp)		
	sw	\$s0,0(\$sp)	}	corpo da função
	add	\$t0,\$a0,\$a1		
	add	\$t1,\$a2,\$a3		
	sub	\$s0,\$t0,\$t1	}	overhead de suporte a funções
	lw	\$s0,0(\$sp)		
	lw	\$t0,4(\$sp)		
	lw	\$t1,8(\$sp)	}	
	addi	\$sp,\$sp,12		
	jr	\$ra		

Pilha para Argumentos e Dados



Mais de 4 Argumentos?

- É comum escrevermos procedimentos em linguagens de alto nível que utilizam mais do que quatro argumentos;
- Possuímos apenas quatro registradores!
- Solução:
 - Utilizamos a pilha para acomodar os argumentos adicionais!

Porque Dois Valores de Retorno?

- Considere o seguinte fragmento de código:

```
addi      $t1,$zero,4294967295
```

```
addi      $t2,$zero,2
```

```
mul       $t3,$t1,$t2
```

- O resultado é maior que 32 bits
- MIPS utiliza dois registradores especiais, \$hi para os 32 bits mais significativos e \$lo para os 32 menos significativos;
- Como retornar esse valor de uma função?

Procedimentos Aninhados

- Problema:
 - \$ra contém o endereço de retorno. Se chamarmos um procedimento a partir de outro procedimento, seu valor será sobrescrito e não será possível retornar da chamada original
 - O mesmo problema se aplica aos registradores de argumentos;
- Solução:
 - Salvar os valores de argumentos e endereço de retorno na pilha!

Exemplo

```
int fat(int n) {  
    if(n<1) return 1;  
    else return n*fat(n-1);  
}
```

```
fat:    addi        $sp,$sp,-8  
        sw          $ra,4($sp)  
        sw          $a0,0($sp)  
        slti        $t0,$a0,1  
        beq         $t0,$zero,L1  
        addi        $v0,$zero,1  
        addi        $sp,$sp,8  
        jr          $ra  
L1:     addi        $a0,$a0,-1  
        jal         fat  
        lw          $a0,0($sp)  
        lw          $ra,4($sp)  
        addi        $sp,$sp,8  
        mul         $v0,$a0,$v0  
        jr          $ra
```

Bibliografia Comentada



- **PATTERSON, D. A. e HENNESSY, J. L. 2014.** *Organização e Projeto de Computadores – A Interface Hardware/Software*. Elsevier/ Campus 4ª edição.



- **HENNESSY, J. L. e PATTERSON, D. A. 2012.** *Arquitetura de Computadores – Uma Abordagem Quantitativa*. Elsevier/ Campus 5ª edição.

Bibliografia Comentada



- **MONTEIRO, M. A. 2001.** *Introdução à Organização de Computadores.* s.l. : LTC, 2001.



- **MURDOCCA, M. J. e HEURING, V. P. 2000.** *Introdução à Introdução de Computadores.* 2000. 85-352-0684-1.

Bibliografia Comentada



- **STALLINGS, W. 2002.** *Arquitetura e Organização de Computadores.* 2002.



- **TANENBAUM, A. S. 2007.** *Organização Estruturada de Computadores.* 2007.