

Universidade Federal de Santa Catarina - Centro Tecnológico - Departamento de Informática e Estatística

INE 5411 - Organização de Computadores

Roteiro do Laboratório 2 - Implementação de Switch/Case

Pré-requisitos para a compreensão e execução do laboratório: Instruções aritméticas, lógicas e de desvio.

1. Objetivo

O objetivo desta aula é estudar uma alternativa eficiente para a geração de código associado à construção *case/switch*, a qual é suportada em linguagens de alto nível contemporâneas (como C e Java) como opção ao aninhamento de desvios condicionais do tipo *if-then-else*. Essa alternativa é baseada no uso de uma “*jump address table*” (JAT), a qual representa um arranjo que armazena os endereços de sequências de instruções das alternativas. Nesse contexto, inicialmente você receberá instruções sobre os fundamentos de uma JAT para, em seguida, gerar um código em linguagem de montagem correspondente a um trecho de programa em linguagem C contendo uma construção *case/switch*. Depois de gerado o código, seu funcionamento deverá ser verificado através de um procedimento de teste.

2. Estudo de caso

No código abaixo (escrito em linguagem C ou Java), uma (e somente uma) dentre cinco alternativas (mais o caso default) de cálculo do valor de *f* é selecionada para ser executada dependendo do valor da variável *k*, onde *k* é um inteiro no intervalo $[0,4]$. A semântica do código pressupõe que, para valores de *k* fora desse intervalo, o caso default deve ser executado. Além disso, a inserção de comandos *break* após o cálculo do valor de *f* denota que as alternativas são mutuamente exclusivas.

```
switch ( k ) {  
    case 0: EXPRESSAO (definida no relatório); break; /* k = 0 */  
    case 1: f = g - h; break; /* k = 1 */  
    case 2: f = g + h + j; break; /* k = 2 */  
    case 3: f = i | h | j; break; /* k = 3 */  
    case 4: f = h & k; break; /* k = 4 */  
    default: f = i - k + 5; break; /* k fora do intervalo [0,4] */  
}
```

Convenções para o exercício

- Adote a seguinte alocação de registradores: (*f, g, h, i, j, k*) -> (\$s0, \$s1, \$s2, \$s3, \$s4, \$s5).
- Atribua os **labels L0, L1, L2, L3, L4 e default** às posições de memória onde são iniciadas as instruções que codificam cada um dos casos de execução (*k=0, k=1, k=2, k=3, k=4 e k fora do intervalo [0,4]*, respectivamente).
- Atribua o **label Exit** à posição de memória que representa a primeira instrução após a execução do trecho do programa.

Procedimento de teste

O procedimento de teste usa a seguinte inicialização de referência para todas as variáveis (exceto *k*): **f=1, g=2, h=4, i=8, j=16**. Essa inicialização é fixada na área de dados globais do programa para facilitar o teste, de forma que apenas o valor de *k* é alterado para cada teste.

- Atribua sucessivamente os valores presentes na tabela abaixo (linha **Valor Inicial k**) à variável *k* através da edição do valor do registrador (não use a memória) e monitore o conteúdo do registrador \$s0. Não se esqueça de reiniciar o programa novamente para cada vez que um valor diferente for atribuído à variável *k*.
- Verifique se os resultados esperados presentes na tabela abaixo (linha **Valor Final f**) são obtidos em \$s0.

| Valor Inicial k | -2 | -1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|-----------------|----|----|----|----|----|----|---|---|---|
| Valor Final f | 15 | 14 | 24 | -2 | 22 | 28 | 4 | 8 | 7 |

Exercício 1:

a) Implemente o trecho de código do estudo de caso em linguagem de montagem do processador MIPS usando uma JAT. O código abaixo ilustra esquematicamente a estrutura do arquivo de programa em linguagem de montagem. Na sequência são apresentadas dicas para a codificação de cada uma das seções:

| | |
|--|--|
| <pre>.data # Seção 1: variáveis f, g, h, i, j # armazenadas em memória # (inicialização) _f: .word 1 #variável f e valor inicial _g: .word 2 _h: .word 4 _i: .word 8 _j: .word 16 # Seção 2: jump address table jat: .word L0 #endereço do label L0 .word L1 .word L2 .word L3 .word L4 .word default #endereço do label default .text .globl main main: # Seção 3: registradores recebem valores # inicializados (exceto variável k) lw \$s0, _f</pre> | <pre>lw \$s1, _g lw \$s2, _h lw \$s3, _i lw \$s4, _j # carrega em \$t4 o endereço-base de jat la \$t4, jat # Seção 4: testa se k está no intervalo # [0,4], caso contrário desvia p/ default ... # Seção 5: calcula o endereço de jat[k] ... # Seção 6: desvia para o endereço que se # encontra armazenado em jat[k] ... # Seção 7: codifica as alternativas de # execução ... Exit: nop</pre> |
|--|--|

Seção 1: Estabelece os valores das variáveis de interesse (exceto k) que devem residir inicialmente na memória.

Seção 2: Descreve o conteúdo da JAT (os endereços das alternativas de execução são aqui representados pelos rótulos de L0 a L4 e default).

Seção 3: Descreve a inicialização dos registradores alocados para as variáveis (**exceto k**). Ao final desta seção, a pseudo-instrução `la $t4, jat` carrega em \$t4 o endereço-base de JAT.

Seção 4: Nesta seção, **escreva um trecho de código usando apenas duas instruções nativas** (não use pseudoinstruções) para testar se k está no intervalo apropriado e desviar para o caso default se k estiver fora de tal intervalo.

Seção 5: Calcule, nesta seção, o endereço efetivo de JAT[k]. Lembre-se que cada elemento da JAT é uma palavra que representa um endereço de memória. Além disso, multiplique o valor de k por 4 para obter o deslocamento em bytes em relação ao endereço-base (início) da JAT.

Seção 6: Primeiramente, carregue o conteúdo de JAT[k] em um registrador temporário \$t0. Em seguida, programe um desvio para o endereço armazenado em \$t0.

Seção 7: Codifique as seis alternativas de execução identificadas pelos labels L0, L1, L2, L3, L4 e default.

Restrições: Nas seções de 4 a 7 você deve usar somente instruções nativas. Não armazene o valor de k em memória. Na seção 7, você deve usar o mínimo número de instruções nativas para implementar cada um dos casos.

b) Salve o programa (opção “File -> Save as”).

c) Simule a execução do código (teclas F5 ou F7 ou seus respectivos botões) e verifique seu funcionamento de acordo com o procedimento de teste definido na primeira página. Adapte o código até que os resultados esperados sejam alcançados.

d) Responda às Questões de 1.1 a 1.10 do relatório de aula.

Sugestão: verifique as opções disponíveis no menu “Settings” para lhe auxiliar nas respostas.

Sugestão: deixe a primeira instrução de cada “case” na mesma linha do label.