

Laboratório 7 - Tratamento de Exceção

considere a motivação alheia, questione tudo.

Sadron

17 de agosto de 2015

1 Fibonacci

Seta \$s0 com o endereço base dos registradores de **MMIO**.

```
move $t2, $zero  
lui $s0, 0xffff
```

Habilite as interrupções para entradas de teclado.

```
lw $t2, 0($s0)  
ori $t2, $t2, 0x2  
sw $t2, 0($s0)
```

Pergunta: em que instrução ocorre overflow na série de fibonacci em linguagem de máquina, experimente tratar o processo no momento em que você capta se é overflow ou não, para testar você mesmo, eu posso estar errado.?

Pista: $\text{return fib}(n-1) + \text{fib}(n-2)$

The screenshot shows a MIPS simulator interface. The assembly code window displays the following instructions:

```

36: loop: beq $t0,$zero,cont # - Neste loop, o número da série de Fibonacci é cal...
37: add $t3,$t2,$t1
38: move $t1,$t2
39: move $t2,$t3
40: addi $t0,$t0,-1
41: j loop
42: cont: sw $t1,8($s1) # - Grava o
44: move $s6,$t1
45: ori $a0,$zero,0x30 # / - 4 instru

```

The registers window shows the following values:

Register	Value
\$8 (vaddr)	8
\$12 (status)	0x0000ff13
\$13 (cause)	0x00000030
\$14 (epc)	0x00400064

The MMIO simulator window shows the display output:

```

I
8
0x00000015
e
0x00000000
800x00000000

```

2 Tratador de Exceções

Etapa 1 - Salvamento de contexto: Os registradores usados pelo tratador são preservados, o que não pode ser feito usando a pilha.

```

lui $k0,0x9000
sw $ra,0($k0)
sw $at,4($k0)
sw $t0,8($k0)
sw $t1,12($k0)

```

Etapa 2 - Decodificação do registrador de causa: O objetivo é descobrir a causa da exceção para realizar a ação apropriada (por exemplo, se for uma interrupção, fazer a leitura e o processamento do valor digitado).

```

mfc0 $k1, $13
srl $t0, $k1, 0x2
andi $t0, $t0, 0xf

```

Etapa 3 – Tratamento de interrupção: Leitura da posição de memória 0xFFFF0004 e processamento do valor digitado. Isso é feito chamando um procedimento `lechar`.

```

bne $t0, $zero, _continua
jal lechar
j done

```

Etapa 4 - Tratamento de **overflow**: O cálculo de um número da série de Fibonacci pode resultar em **overflow** em função do índice digitado. Nesse caso, é preciso tratar o **overflow**, o que deve ser feito imprimindo um “I” no **display** do usuário e reiniciando o programa.

```

add $t1, $zero, 12
bne $t0, $t1, done

la $t1, main
mtc0 $t1, $14

```

Etapa 5 - Preparação do sistema para novas exceções: Consiste na atualização dos registradores Cause e Status para habilitar o tratamento de novas exceções que vierem a ser detectadas.

```

mfc0 $k0, $14
addiu $k0, $k0, 4
mtc0 $k0, $14
mtc0 $0, $13
mfc0 $k0, $12
ori $k0, $k0, 0x1
mtc0 $k0, $12

```

Etapa 6 - Restauração de contexto: Recuperação dos valores de registradores preservados.

```

lui $k0, 0x9000
lw $ra, 0($k0)
lw $at, 4($k0)
lw $t0, 8($k0)
lw $t1, 12($k0)

```

Etapa 7 - Retorno ao fluxo normal de execução ou ao início do programa: No caso da exceção gerada pelo uso do teclado (interrupção), voltar à execução da instrução que deveria ter sido executada. No caso de **overflow**, reiniciar o programa.

```

eret

```