Universidade Federal
do Espírito Santo

CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA

Arquitetura e Organização de Computadores – Turmas 01 e 02 – 2022/2
Prof. Rodolfo da Silva Villaça – rodolfo.villaca@ufes.br

Laboratório V – Conflitos no Pipeline

1. Objetivos

Após completar as atividades deste laboratório, você irá:

- Conhecer como funciona o pipeline da CPU MIPS;
- Verificar o ganho de desempenho da CPU pipeline com relação à CPU monociclo;
- Verificar a influência dos conflitos no desempenho da CPU pipeline;
- Verificar com alguns conflitos podem ser evitados pela reorganização de instruções.

2. Atividades

Para executar esta atividade você poderá utilizar o simulador DrMIPS[1]. Considere o programa *fibonacci.asm* a seguir para executar esta atividade. O programa deveria calcular a famosa sequência de Fibonacci e armazenar os valores de cada passo do algoritmo na memória em *vout* e armazenar o último valor calculado em *res*.

```
.data
num:    .word   10
res:    .word   0
vout:   .space  400

.text
#carregando registradores com valores aleatorios
la $s1, Vout
lw $a0, num
li $t0, 1
li $t1, 1
li $t2, 1

loop:
ble $a0, $t0, end
sw  $t2, 8($s1)
addi $s1, $s1, 4
addi $t0, $t0, 1
addi $14, $0, -2
move $t3, $t2
add  $t2, $t2, $t1
move $t1, $t3
b loop

end:
sw $t2, 8($s1)
sw $t2, res
```

Tarefa 1: Avalie a corretude do código (tem erros!!) e, se necessário, corrija o programa *fibonacci.asm* no DrMIPS[1] com a **CPU Monociclo** (*unicycle.cpu*) para valores de *num* = {10, 20, 30, 40} e preencha a tabela a seguir com a variável *res* no final da execução e a quantidade de instruções executadas.

Dica: Use a ferramenta de geração de estatísticas de execução para auxiliar na sua resposta.

1 https://github.com/brunonova/drmips

num	*res*	*#instrucoes*
10		
20		
30		
40		

Tarefa 2: A seguir você deverá executar o seu código numa **CPU pipeline de 5 estágios**, conforme projeto apresentado no Capítulo 4 do livro texto (usar o modelo **pipeline-no-hazard-detection.cpu** no DrMIPS). Analise o código em busca de possíveis conflitos que causariam problemas na execução no pipeline original, e preencha a tabela a seguir com a variável *res* no final da execução e a quantidade de instruções executadas.

num	*res*	*#instrucoes*
10		
20		
30		
40		

Questões:
a) Quais são estes possíveis os conflitos e a sua causa (dependência dados, mudança de fluxo)?
b) Quais conflitos podem ser resolvidos por reorganização das instruções durante a compilação? Apresente o código reorganizado.

Tarefa 3: Execute **o seu programa** *fibonacci.asm* (**corrigido**) na **CPU pipeline (pipeline.cpu)** do simulador DrMIPS e preencha a tabela abaixo, mostrando: a variável res no final da execução, a quantidade de instruções executadas e o número de bolhas geradas durante a execução do seu programa:

num	*res*	#instruçoes	#bolhas
10			
20			
50			
90			

Questões: Calcule o ganho de desempenho da CPU pipeline em comparação com a CPU monociclo para as mesmas entradas. Justifique sua resposta.