

Laboratório 4

Exercício 1:

- **Abordagem:** Do ponto de vista da lógica, a abordagem foi bastante simples. Bastou seguir o algoritmo fornecido no enunciado do exercício. O principal desafio que enfrentamos foi determinar como converter um número inteiro em um número de ponto flutuante. Até então, estávamos armazenando os valores já convertidos na memória, mas depois de descobrirmos como fazer essa conversão, os processos propostos no exercício tornaram-se mais acessíveis.

Uma vez que o exercício envolvia o uso do coprocessador, tivemos que ajustar nossa abordagem para incorporar os procedimentos em conjunto com o coprocessador. Essa adaptação foi relativamente simples de realizar, especialmente após termos concluído o laboratório 3.

O código inclui dois procedimentos principais. O primeiro procedimento é responsável por solicitar os elementos que serão inseridos nos vetores A e B, e o valor de N representa o tamanho desses vetores. Fizemos uma chamada de sistema (syscall) para ler um valor de ponto flutuante do teclado, e esse valor foi então armazenado no vetor previamente especificado.

O segundo procedimento calcula a média dos valores nos vetores. Ele recebe os valores dos vetores, soma-os e, em seguida, divide pelo valor de N, que representa o número total de elementos. Nesse ponto, foi necessário converter o valor de N em um número de ponto flutuante. Para fazer isso, movemos o valor de N para o coprocessador e, em seguida, o convertemos em ponto flutuante. Depois disso, pudemos realizar a divisão usando os registradores \$f3 e \$f4.

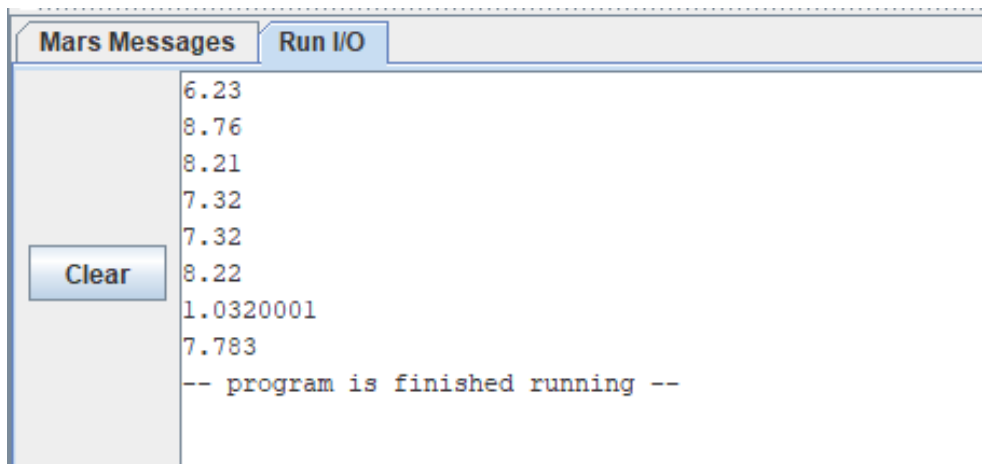
É importante destacar que, após o cálculo da média do vetor A, foi necessário zerar o registrador que armazenava os valores, a fim de calcular a média do vetor B.

No final, conseguimos obter o resultado desejado, embora com mais casas decimais do que as especificadas no exercício.

- **Dificuldades:** O principal desafio que enfrentamos foi a conversão de palavras para ponto flutuante, como mencionado anteriormente, e a lógica das operações entre os registradores do coprocessador. Além disso, tivemos dificuldades com a lógica de zerar os registradores após concluir o cálculo de um vetor e iniciar o cálculo do próximo. Uma das maiores dificuldades foi encontrar uma maneira de definir o tamanho dos nossos vetores como N. Inicialmente, consideramos alocar um espaço pré-definido na memória para os valores, mas posteriormente descobrimos métodos alternativos, mas mantivemos a primeira opção. Um deles permite definir o tamanho do vetor, enquanto o outro envolve o uso de uma syscall para manipular

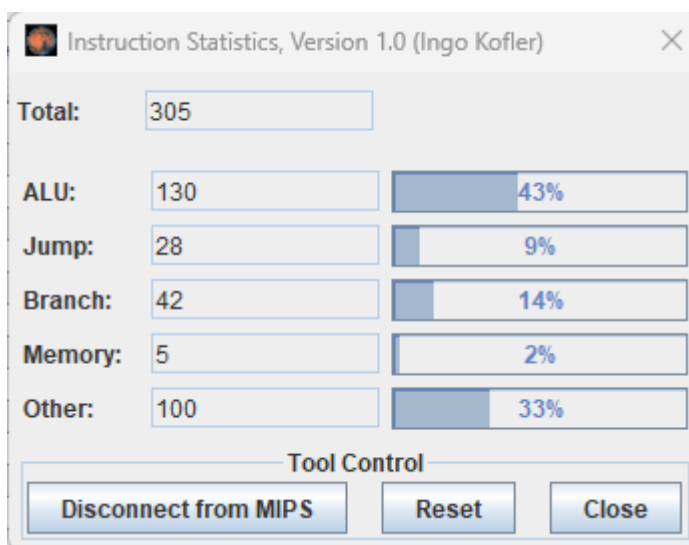
a memória heap do MIPS, embora não tenhamos explorado muito a fundo esse último método.

Terminal depois do cálculo da média:



```
6.23
8.76
8.21
7.32
7.32
8.22
1.0320001
7.783
-- program is finished running --
```

Ferramenta de Desempenho:



Exercício 2:

Neste exercício, enfrentamos um desafio devido à natureza concisa do nosso exercício 1, o que dificultou a implementação de alterações significativas. No entanto, realizamos testes, fazendo ajustes básicos, o que resultou em melhorias modestas no desempenho, um avanço notável.

Inicialmente, para otimizar o desempenho, priorizamos o uso de registradores temporários em detrimento dos registradores de propósito geral. No entanto, isso não produziu resultados significativos.

Em seguida, identificamos uma oportunidade de otimização ao perceber que o valor de N estava sendo convertido para ponto flutuante duas vezes em nosso código, o que poderia afetar o desempenho. Assim, trabalhamos para realizar essa conversão apenas uma vez.

Por fim, buscamos melhorar ainda mais o desempenho ao reduzir o número de instruções que acessam a memória. Observamos que essa abordagem resultou em uma otimização pequena, mas notória, reduzindo a carga na memória e na Unidade Lógica e Aritmética (ULA) durante o processo.

Ferramenta:

