UNIVERSIDADE FEDERAL DE SANTA CATARINA

CAMPUS TRINDADE

INE-DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

INE5411 - ORGANIZAÇÃO DE COMPUTADORES I

Alunos: João Victor Cabral Machado e Pedro Alfeu Wolff Lemos

22204113 **22200373**

Relatório do laboratório 7

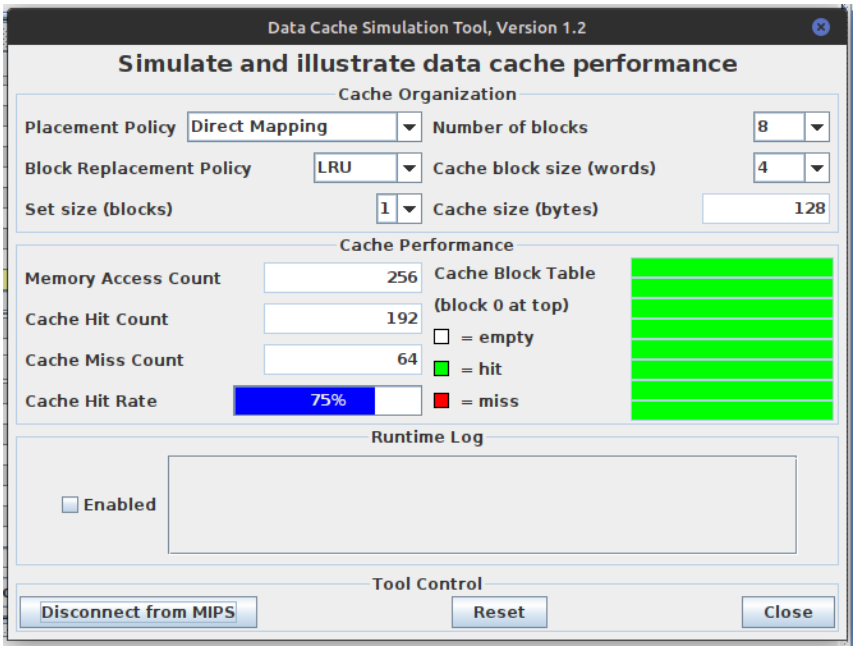
Florianópolis

2023

Questão 1)

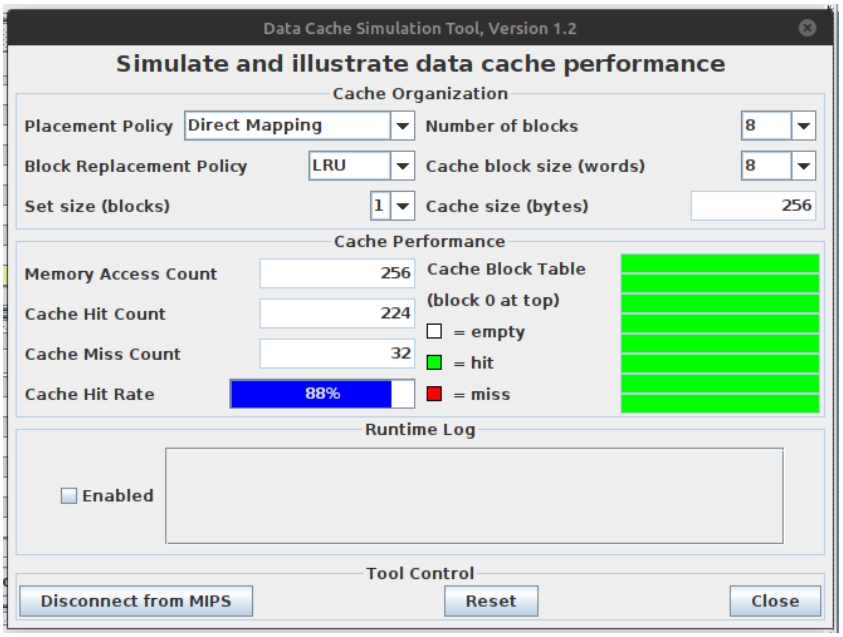
Qual foi o desempenho do cache para este programa?

Resposta: 75%, o erro é sempre no começo de cada word do bloco.

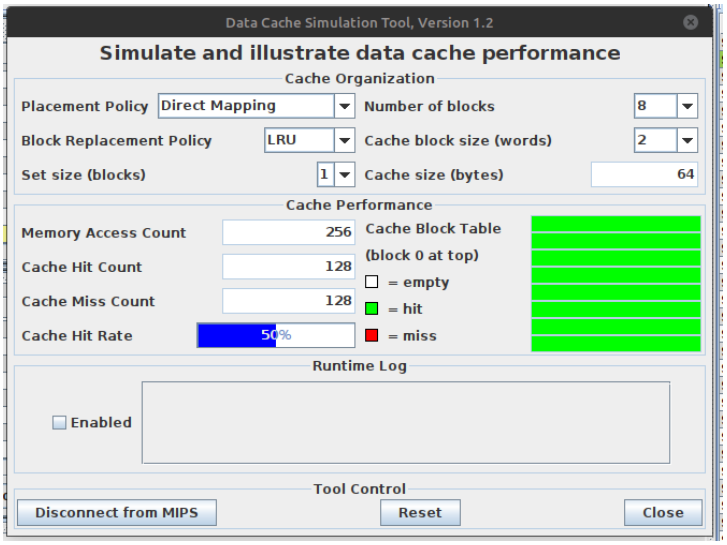


Dada essa explicação (acima), qual será a taxa de acertos se o tamanho do bloco for aumentado de 4 para 8 words

Resposta: 88% ou seja uma falha a cada 7 acertos.

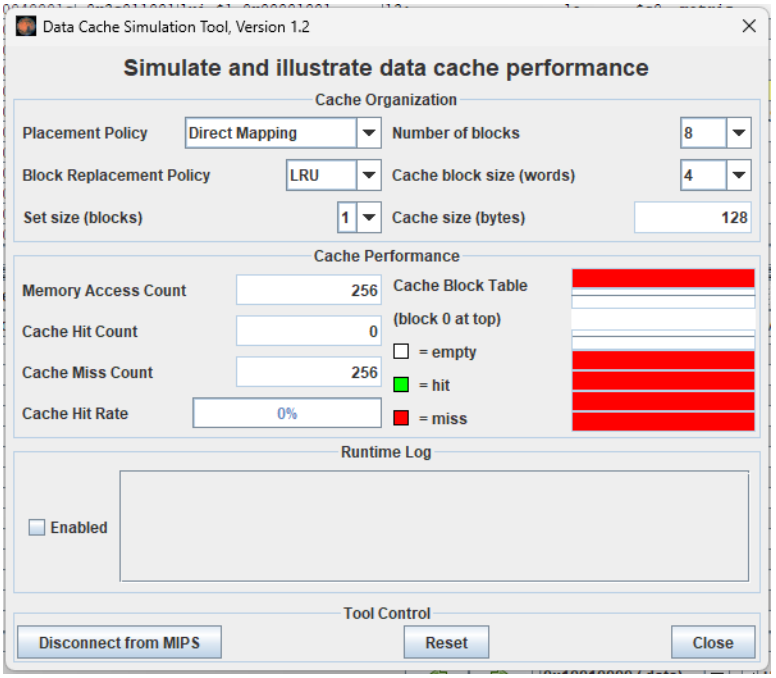


E se for diminuído de 4 words para 2 word
Resposta: 50%, nesse caso a taxa é de um acerto por erro.

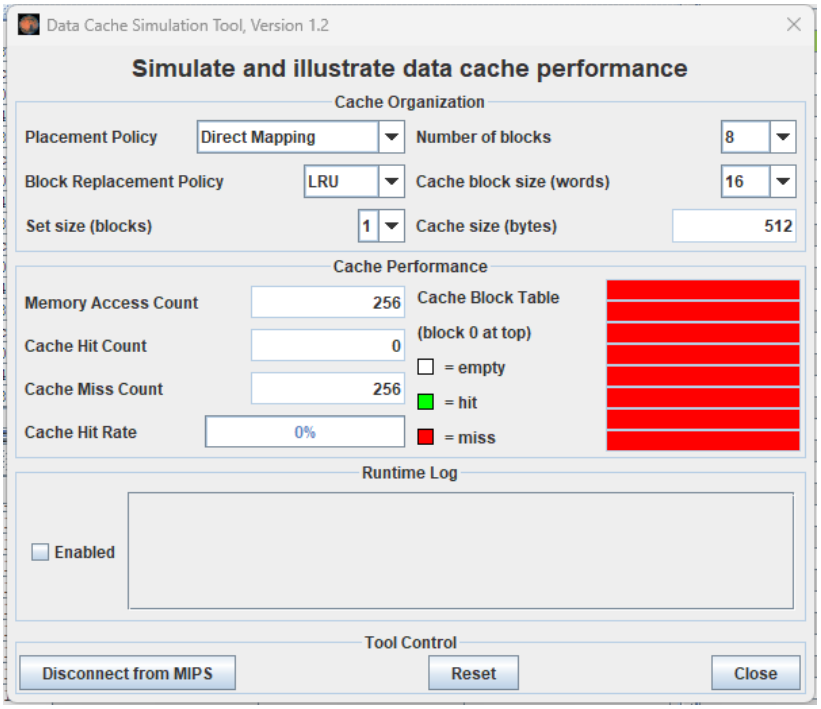


Questão 2)

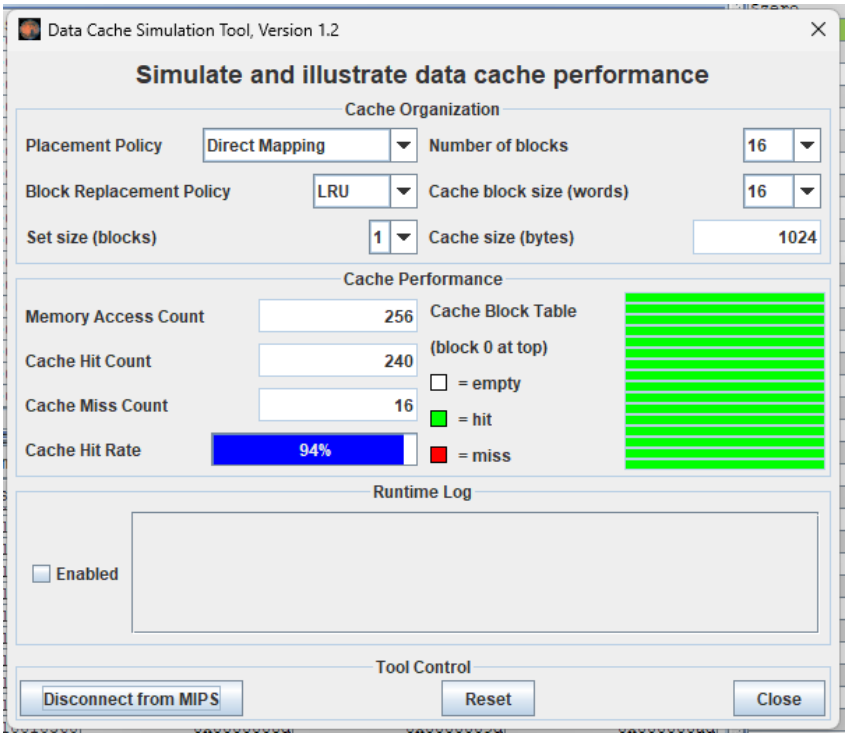
Qual foi o desempenho do cache para este programa?
Resposta: 0%, pois o próximo endereço de memória nunca vai ser o certo, porque ele sempre pula para a próxima coluna.



Qual é o desempenho do cache da instância da ferramenta original, 8x16?
Resposta: Assim como na pergunta anterior, o desempenho é 0%.



Qual é o desempenho do cache da segunda instância da ferramenta 16x16?
Resposta: O desempenho aumenta para 94%. Isso acontece por causa que agora o programa vai armazenar o endereço da memória de toda a matriz após percorrer ela pela primeira vez. Então vamos ter 16 erros (vai ler toda a matriz) e depois vai ser só acertos, pois todos os endereços já foram lidos.



Questão 3)

Considerando as duas implementações que você realizou para o Laboratório 5, qual das duas teve um melhor desempenho?

Resposta: A primeira possui um melhor desempenho, pois ela possui menos erros de acertos, pelo fato de não precisar pular colunas. Isso se deve por causa da maneira que as arquiteturas modernas acessam a memória cache. Os principais princípios por trás disso é:

1) Localidade de Cache. Que basicamente implica que quando um elemento é carregado, os elementos próximos também são carregados pela cache, e isso torna o processo mais eficiente, pois quando estamos acessando uma matriz linha após linha o próximo elemento já vai estar carregado, diferente de coluna por coluna, que o programa vai registrar um miss pois o endereço ainda não está carregado.
2) Métodos de Acesso à Memória. Isso implica que o preenchimento linha após linha resulta em um padrão de acesso à memória mais linear em comparação com o preenchimento coluna após coluna. Isso se alinha mais com a maneira que os sistemas computacionais modernos utilizam a memória, já que ela é normalmente disposta em uma ordem de linha principal.
3) Vetorização. Os processadores modernos utilizam instruções vetorizadas, que múltiplos endereços de memória são acessados de forma simultânea. Então, quando vocẽ acessa ele de maneira contínua, vulgo linha após linha, ela fica mais fácil para o processador realizar essas operações vetoriais.