GABARITO DA AP3 - 2° sem de 2019
ORGANIZAÇÃO DE COMPUTADORES

1. (3,5) Considere uma máquina com arquitetura semelhante à arquitetura vista no curso, que apresente as seguintes especificações:

   - Capaz de endereçar 128 M células de memória principal.
   - Deve possuir um registrador Acumulador, além do RDM (Registrador de Dados da Memória), REM (Registrador de Endereços da Memória), CI (Contador de Instrução) e RI (Registrador de Instrução).
   - O conjunto de instruções de linguagem de máquina deve ter 31 instruções.
   - Cada instrução deve conter um código de operação e um operando como mostrado abaixo, onde o operando indica um endereço de memória e um registrador

Código de operação	Operando

   a) (0,3) Lembrando que $2^7\ 2^{20} = 128$ M, um REM de 16 bits satisfaz os requisitos do sistema? Justifique claramente.

   Sabemos que o Barramento de endereços (BE) determina o tamanho máximo da MP que a cpu pode acessar e que o BE tem o mesmo tamanho que o REM (registrador de endereços da memória). Um REM com 16bits, consequentemente BE também terá 16 bits e o tamanho máximo da MP que a CPU poderá acessar será de $2^{16} = 65536$ células (ou 64K células). Diante do apresentado, um REM com 16 bits é insuficiente para endereçar as 128M células

   O tamnho mínimo do BE deverá ser = $\log_2 128M = \log_2 2^7 2^{10} = \log_2 2^{27} = 27$ bits, assim, um REM que satisfaça os requisitos do sistema deverá ter o tamanho mínimo de 27bits.

   b) (0,3) Lembrando que $2^5 = 32$, instruções de 16 bits satisfazem os requisitos do sistema? Justifique claramente.

   O conjunto de instruções de linguagem de máquina é de 31 instruções. São necessários 5bits para o campo do cód.opr. , $2^5 = 32$, para atender a exigência de 31 intruções.
   O operando deverá ter um tamanho mínimo para qualquer endereço de memória = 27 bits
   Como a instrução é formada de um cód.opr. e um operando, o tamanho da instrução deverá ter o tamanho mínimo = 5bits + 27bits = 32bits.
   Concluíndo, instruções de 16bits não atendem aos requisitos do sistema, necessário instruções com 32bits.

   c) (0,8) Suponha que a máquina seja construída de forma minimalista, ou seja, usando REM e instruções com o menor número possível de bits. Indique o tamanho em bits de cada célula da memória principal, o tamanho do RDM e o barramento de dados de modo que a Unidade Central de Processamento obtenha uma instrução da memória principal realizando somente um acesso à memória principal.

   Para que o barramento de dados (BD) obtenha uma instrução da memória principal realizando

somente um acesso, o BD deverá ter 32 bits, como RDM = BD, RDM também terá 32 bits.

A cada acesso, 1 célula é acessada, como BD tem 32bits, a célula deverá ter 32bits.

d) (0,6) Mais uma vez, suponha que a máquina seja construída de forma minimalista, ou seja, usando REM e instruções com o menor número possível de bits. Calcule o tamanho de RI e CI.

RI (registrador de instruções) deverá ter no mínimo o tamanho de uma instrução, RI = 32 bits

CI (contador de instruções) deverá ter o tamanho mínimo para acessar toda a memória, CI = 27bits

e) (0.3) Calcule a capacidade de armazenamento, em bits, da memória desta máquina.

Capacidade de armazenamento (T) será igual a quantidade de células (N) x o tamanho de cada célula (M),
T = M x N => T = 128M células x 32 bits/célula = 4Gbits ou 512MBytes

f) (0,6) Descreva detalhadamente a execução da instrução **LDA Op.** nesta máquina. A instrução **LDA Op.** carrega o acumulador com o conteúdo da célula de memória cujo endereço é Op.

Passo 1: A CPU coloca no REM o valor do operando (REM ← Op) e é disponibilizado no barramento de endereço
Passo 2: A CPU aciona pelo barramento de controle o sinal de leitura de memória
Passo 3: A memória coloca o valor no barramento de dados, correspondente ao endereço contido no barramento de endereços, a seguir chega no RDM da CPU (RDM ← MP(Op))
Passo 4: O valor armazenado no RDM é transferido para o Acumulador ACC ← RDM (ou ACC ← MP(Op))
Passo 5: CI é incrementado (CI ← CI+1) para apontar para a próxima instrução a ser lida.

g) (0,6) Descreva detalhadamente a execução da instrução **MUL Op.** nesta máquina. **MUL Op.** multiplica o conteúdo da célula de memória cujo endereço é **Op**. pelo conteúdo do acumulador e armazena o resultado na memória no endereço **Op**.

Passo 1: A CPU coloca no REM o valor do operando (REM ← Op) e é disponibilizado no barramento de endereço
Passo 2: A CPU aciona pelo barramento de controle a leitura de memória
Passo 3: A memória coloca o valor no barramento de dados, e por consequência no RDM da CPU (RDM ← MP(Op))
Passo 4: A CPU executa a multiplicação do valor recebido com o contido no acumulador armazenando o resultado no acumulador; ACC ← ACC × RDM (ou ACC ← ACC × MP(Op))
Passo 5: CI é incrementado (CI←CI+1) para apontar para a próxima instrução a ser lida

2. (1,5) Considere uma máquina que utiliza 8 bits para representar números em ponto

fixo e em ponto flutuante.

a) (0,8) Mostre a representação de -1,5 utilizando-se a representação ponto flutuante precisão simples (1 bit de sinal, 4 bits para expoente em excesso de 7, 3 bits para mantissa)

$-1,5_{10} = -1,1_2 = -1,1 \times 2^{+0}$

Sinal = negativo = 1
Expoente = + 0 + 7 = 7_{10} = 0111_2
Mantissa = ,1

Concluíndo: 1 0111 100

b) (0,7) Suponha que um programador desavisadamente acessou o conjunto de bits obtido no item anterior pensando que se tratava de
   i. (0,3) um inteiro sem sinal

$$10111100_2 = 2^7 + 2^5 + 2^4 + 2^3 + 2^2 = 188$$

   ii. (0,4) um inteiro utilizando a representação em complemento a 2

$$10111100_2 = -2^7 + (2^5 + 2^4 + 2^3 + 2^2) = -68$$

Em cada caso, o programador mandou imprimir na tela o conteúdo processado. O que o programador imprime na tela, na base 10, em cada um dos casos acima? Os valores impressos são maiores, menores, ou iguais ao valor original?

O valor do item b.i é maior (188 > -1,5), já o do item b.ii é menor (-68 < -1,5)

3. (2,5) Quais são os mapeamentos existentes para armazenamento e recuperação de dados (instruções) na memória cache? Explique em detalhes cada um deles.

A memória é dividia em blocos. Cada bloco é constituído de um número K de células. A memória cache é dividida em linhas, onde cada linha possui o mesmo tamanho de um bloco. Um bloco, quando transferido da memória principal, ocupa uma linha da cache.

Ao requisitar uma célula da MP, primeiro é feita a busca na cache e caso a encontre (cache hit), o conteúdo da palavar é enviado para a cpu, entretanto, caso não a encontre (cache miss), será feita a transferência do bloco que contém o endereço solicitado da MP para a cache e daí para o processador.

O número de linhas da cache é bem inferior ao número de blocos da memória principal, diante dissso é necessário implementar algoritmos para mapaear os blocos da memória principal nas linhas da cache e um mecanismo para a localzar e verificar se está na cache. Os mapeamentos são: Direto, totalmente associativo e associativo por conjunto.

No mapeamento direto, cada bloco da MP é mapeado (direcionado) a apenas uma linha da cache. Além do espaço para o bloco que é transferido da MP, cada linha da cache possui um campo que recebe a tag que identifica o bloco. O endereço da MP, requsitado pela cpu, é dividido em 3 campos: o primeiro corresponde a tag (validador), depois o número da linha da cache e, por último, o campo com a posição da palavra dentro do bloco. A implementação deste mapeamento é

mais fácil, pois para ter um cache hit basta que a tag da linha seja igual ao do campo tag do endereço da MP. A linha da cache, em que há a busca, é única e sua posição está no 2º campo da MP. A palavra encontrada, cuja posição é definida pelo 3º campo do endereço da MP, é enviada à cpu. A vantagem, mencionada anteriormente, está na facilidade de implementação por hardware. A desvantagem está na pouca flexibilidade de alocação do bloco na cache.

No mapemento totalmente associativo, cada bloco da MP pode ocupar qualquer linha da cache. Similar ao mapeamento direto, cada linha da cache tb possui um campo para receber a tag, que neste método armazena o número do bloco. O endereço da MP, requsitado pela cpu, agora é dividido em 2 campos: o primeiro sendo a tag (validador) e em seguida o campo com a posição da palavra dentro do bloco. A implementação deste mapeamento é mais dificil, pois há a necessidade de implementar um loop (ou uma comparação paralela) para procurar linha a linha da cache a que possui a tag desejada. Caso encontre (cache hit), a palavra (posição definida pelo 3º campo do endereço da MP) requisitada é enviada à cpu. A vantagem, ao contário do mapeamento anterior, está na flexibilidade de alocação e a devantagem está na implementação e no tempo gasto na busca mencionada.

No mapeamento associativo por conjuntos temos uma combinação das vantagens das técnicas anteriores. Neste método, as linhas da cache são dispostas em conjuntos. Cada bloco da MP poderá ocupar qualquer linha de um conjunto. O endereço da MP, requsitado pela cpu, é dividido em 3 campos: o primeiro corresponde a tag (validador), em seguida o número do conjunto e, finalmente, o campo com a posição da palavra dentro do bloco. Para se ter um cache hit, a busca agora é feita linha a linha apenas no conjunto apontado no 2º campo do endereço da MP.

3. (2,5) Explique em detalhes todos os modos de endereçamento de dados, com vantagens, desvantagens e aplicações.

Abaixo seguem os modos de endereçamento abordados em aula:

Imediato: *O campo operando contém o dado, desta forma o dado é transferido da memória juntamente com a instrução.*
Vantagem: Rapidez na execução da instrução, pois não requer acesso à memória principal, apenas na busca da própria instrução.
Desvantagem. Limitação do tamanho do campo operando das instruções reduzindo o valor máximo do dado a ser manipulado. Trabalho excessivo para alteração de valores quando o programa é executado repetidamente e o conteúdo das variáveis serem diferentes em cada execução.

Direto: *O campo operando da instrução contém o endereço onde se localiza o dado.*
Vantagem. Flexibilidade no acesso a variáveis de valor diferente em cada execução do programa
Desvantagem. Limitação de memória a ser usada conforme o tamanho do operando.

Indireto: *O campo de operando contém o endereço de uma célula, sendo o valor contido nesta célula o endereço do dado desejado.*
Vantagem: Usar como "ponteiro". Elimina o problema do modo direto de limitação do valor do endereço do dado. Manuseio de vetores (quando o modo indexado não está disponível).
Desvantagem: Muitos acesso à MP para execução, pelo menos 2 acessos à memória principal.

No modo indexado: *consiste em que o endereço do dado é a soma do valor do campo operando (que é fixo para todos os elementos de um dado vetor) e de um valor armazenado em um dos registradores da UCP (normalmente denominado registrador índice).*
Vantagem: Rapidez de execução das instruções de acesso aos dados, visto que a alteração do endereço dos elementos é realizada na própria UCP.
Desvantagem: a complexidade da instrução.

No modo de endereçamento base mais deslocamento *o endereço é obtido da soma do campo de deslocamento com o conteúdo do registrador base. Este modo de endereçamento tem como*

principal objetivo permitir a modificação de endereço de programas ou módulos destes, bastando para isso alterar o registrador base.
Vantagem: Reduz o tamanho das instruções e facilita o processo de relocação de programas.
Desvantagem: a complexidade da instrução, como no modo indexado.