Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Organização de Computadores
GABARITO- AP1 1° semestre de 2011.

Nome –
Assinatura –

Observações:

1. Prova sem consulta e sem uso de máquina de calcular.
2. Use caneta para preencher o seu nome e assinar nas folhas de questões e nas folhas de respostas.
3. Você pode usar lápis para responder as questões.
4. Ao final da prova devolva as folhas de questões e as de respostas.
5. Todas as respostas devem ser transcritas nas folhas de respostas. As respostas nas folhas de questões não serão corrigidas.

1. (2,0) Um computador possui uma capacidade máxima de memória principal de 1 Giga Bytes (2 elevado a 30 bytes), organizados em células de 32 bits:

a) Qual é o maior endereço em decimal desta memória (pode deixar a conta indicada)?

Resposta:

Tamanho da memória (T) = 1Gbytes = 8Gbits ou 2^{33} bits
Tamanho da célula (M) = 32 bits/célula ou 2^5 bits/célula:
T = M x N => N = ? => N = T / M => 8Gbits / 32bits
=> N = 2^{33} / 2^5 bits/célula => N = 2^{28} células
Maior endereço (em decimal) = N - 1 = 2^{28} – 1 = ***268.435.455***

b) Qual é o tamanho do barramento de endereços deste sistema?

Resposta:

O Tamanho deste barramento será o suficiente para endereçar todas as células da memória (N).
O tamanho do barramento corresponderá ao valor de ***e*** *em $2^e = N$*
$2^e = N$ => $2^e = 2^{28}$ => e = 28, portanto,
barramento de endereços = 28 bits

c) Quantos bits podem ser armazenados no RDM e no REM?

Resposta:

Tamanho do REM = tamanho do barramento de endereços
REM = 28 bits
Tamanho do RDM = tamanho do barramento de dados e terá de ser no mínimo o tamanho de uma célula
RDM = 32 bits

d) Qual é o número máximo de bits que pode existir na memória?

Resposta:

O total de bits da memória será igual a T
Tamanho da memória (T) = 1Gbytes = 8Gbits ou 2^{33} bits = 8.589.934.592bits

2. (2,0) Considere uma máquina que possa endereçar 1 Giga Bytes de memória física, utilizando endereço referenciando byte, e que tenha a sua memória organizada em blocos de 4 K bytes. Ela possui uma memória cache que pode armazenar 2 K blocos, sendo um bloco por linha (ou quadro). Mostre o formato da memória cache, indicando os campos necessários (tag, bloco) e o número de bits para cada campo, e o formato de um endereço da memória principal, indicando os bits que referenciam os campos da cache, para os seguintes mapeamentos:

Resposta:

a) Mapeamento direto.

Memória Principal

⇒ *Tamanho da memória (em bytes) = 1Gbytes, como 1 célula referencia a 1 byte, temos N = 1G células*

⇒ *Será organizada em blocos de 4K bytes, como 1 célula = 1 byte, temos cada bloco = 4K células, K = 4K*

⇒ *Sendo N o tamanho endereçável da memória, e K a quantidade de células por blocos, 1. temos:*
N = 1G células e K = 4K células / blocos o total de blocos da MP (B) será:
Total de blocos: B = N / K => B = 1G células / 4K células/bloco =>
B = 256 K blocos

Memória Cache
OBS: A quantidade de células/bloco tem de ser igual a da MP.

⇒ *Tamanho da memória cache (em blocos ou linhas) => Q = 2K blocos*

⇒ *Tamanho da memória cache em células = Q x K =*
2K blocos x 4Kcélulas/bloco = 8M células
Cada célula possui 1 byte = 8bits, então,
como a cache possui 8M células x 8bits, que totaliza 64M bits

linha	válido	tag	Conteúdo (bloco)
0	1 bit	7 bits	4096 bytes
1			
2			
3			
4			
5			
.....			
Q - 2			
$2^{11} - 1$			

Endereço da MP:
Para endereçarmos toda a MP precisamos da seguinte quantidade de bits (E)
sendo $N = 2^E$ => N = 1G células => $N = 2^{30}$ => E = 30 bits

Composição do endereço em função da memória cache
=> tag = B / Q = 256K / 2K = 128 = 2^7 = 7 bits
=> nº da linha: Q = 2K linhas ou quadros (máximo) => 2^{11} => 11bits
=> células por bloco: 4K células por bloco = 2^{12} => 12bits

30bits

Tag = 7 bits	*No. Linha =11 bits*	*Célula no bloco=12 bits*

b) Mapeamento totalmente associativo.

Memória Principal
=> N = 1G células
=> K = 4K células/bloco
=> B = 256 Kblocos

Memória Cache
OBS: A quantidade de células/bloco tem de ser igual a da MP.
=> Q = 2K blocos
=> Tamanho da memória cache = 8 Mcélulas ou 8Mbytes ou 64Mbits

linha	válido	tag	Conteúdo (bloco)
0	1 bit	18 bits	4096 bytes
1			
2			
3			
4			
5			
.....			
Q - 2			
$2^{11} - 1$			

Endereço da MP = 30 bits

Composição do endereço em função da memória cache
=> tag = B = 256K = 2^{18} => tag = 18bits
=> células por bloco: 4K células por bloco = 2^{12} => 12bits

30bits

Tag = 18bits	*Célula no bloco=12bits*

c) Mapeamento associativo por conjunto, onde cada conjunto possui duas linhas, cada uma de um bloco.

Memória Principal
=> N = 1G células
=> K = 4K células/bloco
=> B = 256K blocos

Memória Cache
OBS: A quantidade de células/bloco tem de ser igual a da MP.
=> Q = 2K blocos
=> Tamanho da memória cache = 2K células
=> 1 conjunto = 2 linhas (ou quadros) =>

Total de conjuntos => C = 2K blocos / 2 => ***C = 1K conjuntos***

linha	válido	tag	Conteúdo (bloco)
0	1 bit	8 bits	4096 bytes
1			
2			
3			
4			
5			
.....			
Q - 2			
2^{11} - 1			

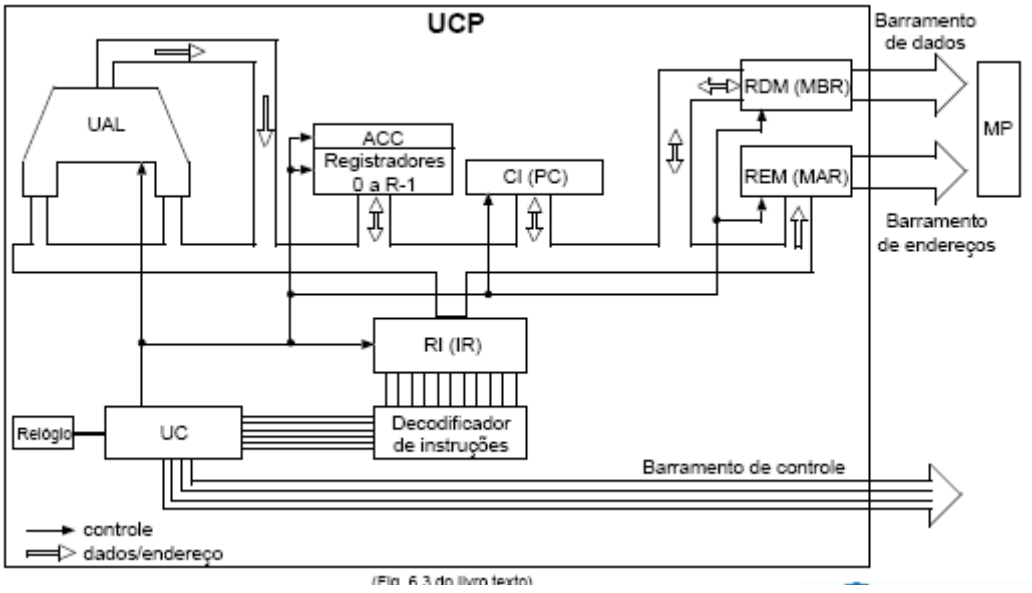
Endereço da MP = 30 bits

Composição do endereço em função da memória cache
=> tag = B / C = 256K / 1K = 256 = 2^8 => tag = 8 bits
=> nº do Conjunto: Q = 1K linhas ou quadros (máximo) => 2^{10} => 10 bits
=> células por bloco: 4 K células por bloco = 2^{12} => 12bits

30 bits

Tag = 8 bits	*No. Conjunto =10bits*	*Célula no bloco=12bits*

3. (2,0) Considere o sistema apresentado em aula mostrado na figura abaixo.



Descreva **detalhadamente** a execução das instruções **STR Op.** e **JN Op**., indicando como o Registrador de Instrução (RI), Contador de Instrução (CI), Acumulador (ACC), Registrador de Dados da Memória (RDM), Registrador de Endereços da Memória (REM), Unidade Aritmética Lógica (UAL) e Barramento de controle, de dados e de endereços são utilizados na execução destas instruções. Lembre-se que a instrução STR Op., quando

executada, carrega o conteúdo do Acumulador na memória cujo endereço é Op. e a instrução JN Op., quando executada, carrega CI com o valor de Op. se o conteúdo do Acumulador é menor que zero, e caso contrário carrega CI com CI+1.

Resposta:

STR Op

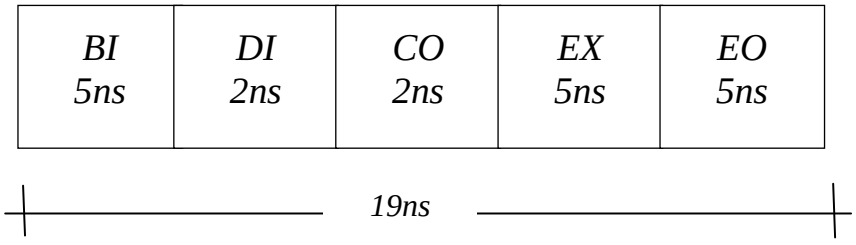
a) RI <- (CI)
b) CI <- CI + 1
c) Decodificação do código de operação
d) Busca do operando na memória
- O valor do campo operando = Op é transferido para o REM
- O conteúdo do Acumulador (ACC) é transferido para o RDM (RDM ← ACC)
- A Unidade de Controle ativa a linha WRITE do barramento de controle
- O conteúdo do REM é transferido para o barramento de endereços
- O conteúdo do RDM é transferido para o barramento de dados
- A memória grava o dado recebido através do barramento de dados no endereço que consta no barramento de endereços (REM) ← RDM, ou melhor, (Op) ← ACC

JN Op

a) RI <- (CI)
b) Decodificação do código de operação
c) A Unidade de Controle emite sinal para transferir conteúdo do acumulador para UAL
UAL <- ACC
d) UAL executa operação de comparação
d.1) Se Resultado = verdadeiro, isto é, ACC < 0
CI <- Op
d.2) Se Resultado = Falso, ou seja, ACC >= 0
CI <- CI + 1

4. (2,5) Considere uma máquina que pode ter seu ciclo de busca e execução de uma instrução dividido em 5 estágios totalmente independentes: Busca de Instrução (BI), Decodificação (DI), Cálculo de Endereços de Operandos (CO), Execução (EX) e Escrita de Operandos(EO). Cada um dos estágios BI, EX e EO possui a duração de 5 ns e cada estágio DI e CO tem duração de 2 ns. Cada instrução desta máquina precisa executar os 5 estágios que serão sempre executados na seqüência BI, DI, CO, EX e EO.

a) (0,2) Uma implementação desta máquina foi realizada de modo que cada instrução deve ser completamente realizada **em um único ciclo de relógio.** Calcule **a duração do ciclo de relógio** que esta implementação deve possuir**.** Lembre-se que todas as instruções necessitam dos 5 estágios.
Resposta:

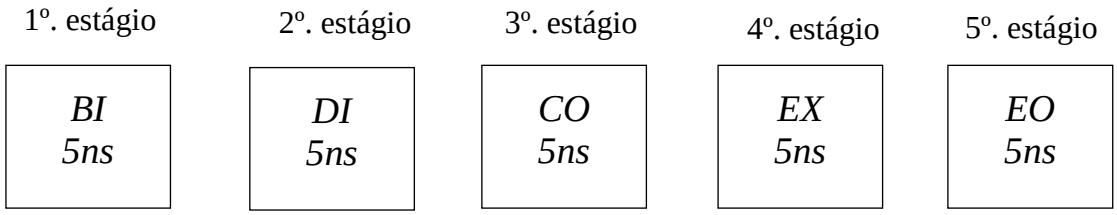


*Ciclo de relógio para execução de uma instrução (sem pipeline) = 5ns + 2ns + 2ns + 5ns + 5ns = **19ns***

b) (0,5) Como cada estágio é independente um do outro, deseja-se implementar uma **nova** arquitetura utilizando-se um pipeline de 5 estágios. Nesta nova implementação **cada**

estágio do pipeline deve ser executado em um ciclo de relógio. Calcule **a duração do ciclo de relógio** que esta implementação pipeline deve possuir.

Resposta:



Ciclo de relógio será igual ao tempo para execução do estágio com maior tempo de execução = 5**ns.**

c) (0,5) Considere um programa que necessita executar 100 instruções. Calcule o tempo de execução deste programa na máquina do item **a** e na máquina do item **b**.

Resposta:

Seja Tex = tempo de execução de uma instrução = número de estágios x ciclo de relógio (determinado nos itens anteriores)

Para o item a (sem pipeline) :

Tex = 1 estágio x 19ns = 19ns

Ttotal = 100 instruções x Tex = ***1.900ns***

Para o item b (pipeline: 5 estágios) :

Tex = 5 estágios x 5ns = 25ns

Ttotal = Tex + 99 x tempo de 1 estágio

Ttotal = 25ns + 99 x 5ns = ***520ns***

d) (0,5) Considere um programa que necessita executar 10000 instruções. Calcule o tempo de execução deste programa na máquina do item **a** e na máquina do item **b**.

Resposta:

Seja Tex = tempo de execução de uma instrução = número de estágios x ciclo de relógio (determinado nos itens anteriores)

Para o item a (sem pipeline) :

Tex = 1 estágio x 19ns = 19ns

Ttotal = 10.000 instruções x Tex = ***190.000ns ou 190µs***

Para o item b (pipeline: 5 estágios) :

Tex = 5 estágios x 5ns = 25ns

Ttotal = Tex + 9.999 x tempo de 1 estágio

Ttotal = 25ns + 9.999 x 5ns = ***50.020ns***

e) (0,8) Compare as diferenças dos tempos de execução obtidos pela maquina do item a e a do item b para os programas dos itens c e d.

Resposta:

Em ambos itens c e d, a máquina que consta do item b (com pipeline) tem um tempo de execução de uma única instrução maior que o do item a, mas para uma sequência de instruções, como a de 100 ou 10000 instruções, a máquina do item b (com pipeline) passa a ter um desempenho superior a máquina do item a (sem pipeline), pois o tempo total de execução do programa se torna menor. Também podemos observar que obtemos um ganho maior de desempenho da arquitetura com pipeline quando executamos um número maior de instruções. Para o programa que executa 100 instruções, obtemos um tempo de

execução para a máquina pipeline igual a 27,34% do tempo obtido na máquina sem pipeline e na execução de1000 instruções obteve-se na máquina pipeline um tempo de execução igual a 26,33% do tempo obtido na máquina sem pipeline.

5. **(1,5) Explique como funciona uma Unidade Central de Processamento (UCP) cujo controle é realizado por hardware e uma UCP com controle por microprograma.**

Resposta:

A unidade de controle implementada por hardware consiste em circuitos combinatórios. Os sinais lógicos de entrada na unidade são transformados em um conjunto lógico de sinais que controlam a execução da instrução. Para implementar a unidade, necessita-se derivar, para cada sinal de controle a ser gerado para que cada instrução seja executada de forma correta, uma expressão booleana que define esse sinal em função dos sinais de entrada referentes à instrução. As expressões booleanas são implementadas com circuitos combinatórios.

A unidade de controle microprogramada é projetada de modo a executar as microinstruções que compõem uma instrução. Ela é composta por: Memória de controle, Contador de microprograma e Sequenciador.

A Memória de controle armazena as microinstruções que compõem uma instrução e o Contador de microprograma armazena a localização da próxima microinstrução a ser executada. O Seqüenciador é o componente que controla a sequência de execução das microinstruções, informando o local da próxima microinstrução que deve ser executada e armazenada no Contador de microprograma.