Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Organização de Computadores
AP1 1° semestre de 2013..

Nome –

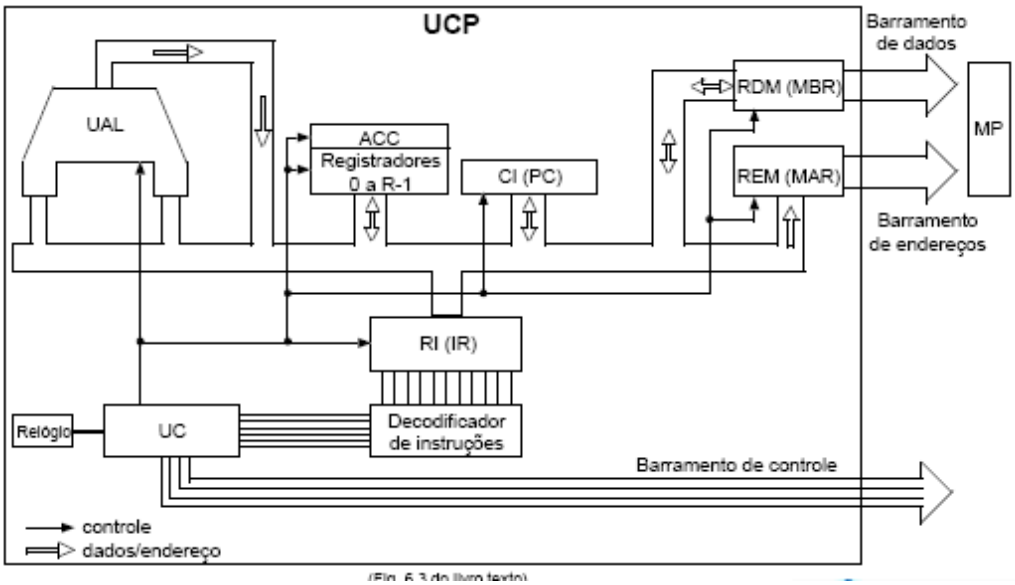
Assinatura –

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) Considere o sistema estudado em aula e mostrado na figura abaixo:



Descreva **detalhadamente** a execução das instruções **SUB Op.** e **JP 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 SUB Op., quando

executada, subtrai o conteúdo da memória cujo endereço é Op. do conteúdo do acumulador e a instrução JP Op., quando executada, carrega CI com o valor de Op. se o conteúdo do Acumulador é maior que zero, e caso contrário carrega CI com CI+1.

SUB Op

a) RI <- (CI)
b) CI <- CI + 1
c) Decodificação do código de operação
- recebe os bits do código de operação
- produz sinais para a execução da operação de subtração
d) Busca do operando na memória
- A UC emite sinais para que o valor do campo operando (Op) seja transferido para a REM
- A UC ativa a linha READ do barramento de controle
- Conteúdo de memória do endereço (Op) é transferido para RDM
e) Execução da operação
- Dados para a operação de subtração são transferidos para a UAL:
UAL(a) <- ACC e UAL(b) <- RDM (a) e (b) são as entradas da UAL
- UAL executa a operação de subtração: UAL(a) - UAL(b) , ou seja, ACC - RDM
- ACC recebe o resultado: ACC <- ACC - RDM

JP Op

a) RI <- Instrução lida
b) Decodificação do código de operação
- recebe os bits do código de operação
- produz sinais para a execução da operação de salto condicional
c) UC emite sinal para transferir conteúdo acumulador para UAL
UAL <- ACC
d) Executa operação de comparação
d.1) Se Resultado = verdadeiro, isto é, ACC > 0
CI <- Operando (CI <- Op)
d.2) Se Resultado = Falso, ou seja, ACC <= 0
CI <- CI + 1
e) Inicia o procedimento de leitura da instrução contida no endereço que consta em CI

2) (2,0) 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 X ns e cada estágio DI e CO tem duração de (X-2) ns. Cada instrução desta máquina precisa executar os 5 estágios que serão sempre executados na seqüência BI, DI, CO, EX e EO.

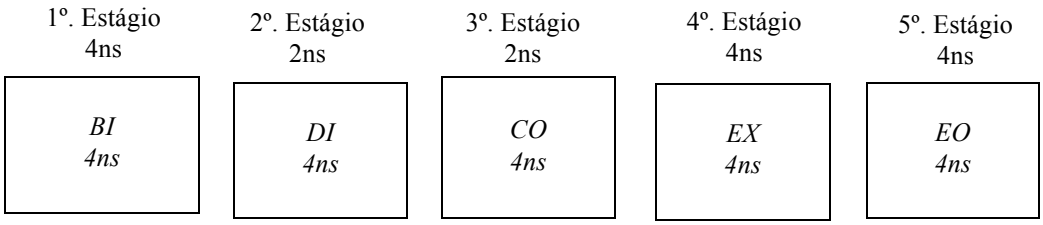
a) (0,2) Uma implementação desta máquina foi realizada de modo que cada instrução deve ser completamente realizada **em um único ciclo de relógio.** Na execução de um determinado programa P, verificou-se que foram executadas 1000 instruções e o programa foi executado em 16000 ns. Calcule **a duração do ciclo de relógio** que esta implementação possui**.**

BI *Xns*	*DI* *(X-2)ns*	*CO* *(X-2)ns*	*EX* *Xns*	*EO* *Xns*

Ciclo de relógio para execução de uma instrução (sem pipeline) =
*X ns + (X-2) ns + (X-2) ns + X ns + Xns = **(5X – 4) ns***

Total de instruções = 1000, e tempo total para execução = 16.000ns,
Tempo para execução de 1 instrução = 16.000 / 1.000 = 16ns
Cada instrução é executada em um ciclo de relógio,
então (5X – 4)ns = 16ns => X = 4ns
Concluíndo, os estágios BI, EX, EO terão duração, cada um de 4ns, DI e CO de 2ns

b) (0,6) 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. Lembre-se que todas as instruções necessitam dos 5 estágios.



*Ciclo de relógio será igual ao tempo para execução do estágio com maior tempo de execução = 4 **ns.***

c) (0,6) Com o valor encontrado no item **b**, calcule o tempo de execução do programa P do item **a** na máquina com pipeline do item **b**.

Seja Tex = tempo de execução de uma instrução = número de estágios x ciclo de relógio
Para o item b (pipeline: 5 estágios) :
Tex = 5 estágios x 4ns = 20ns
Ttotal = Tex + 999 x tempo de 1 estágio
*Ttotal = 20ns + 999 x 4ns = **4016ns***

d) (0,6) Indique em qual das duas máquinas o programa P é executado mais rapidamente e calcule o ganho de desempenho obtido calculando a divisão do maior tempo de execução pelo menor tempo de execução.

O programa P será executado mais rapidamente na máquina do item b (pipeline de 5 estágios).
Ganho de desempenho = 16.000 / 4.016 ns. O tempo de execuçãoda máquina sem pipeline (item a) será 3,98 vezes maior em relação àquela que tem pipeline (item b).

3. (2,0) Explique detalhadamente como funciona uma Unidade Central de Processamento (UCP) com controle por microprograma, e explique os dois métodos utilizados para formatar e usar uma microinstrução: microinstruções verticais e microinstruções horizontais.

Em uma arquitetura microprogramada, a unidade de controle é especificada por um microprograma que consiste de uma seqüência de instruções de uma linguagem de microprogramação. Estas instruções são muito simples e especificam microoperações. Uma unidade de controle microprogramada é implementada com circuitos lógicos e é capaz de seguir uma seqüência de microinstruções gerando sinais de controle para que cada uma delas seja executada. Os sinais de

controle gerados por uma microinstrução são usados para causar transferências de dados entre registradores e memória e execução de operações pela ULA.

As microinstruções horizontais tem como característica ter funções distintas para cada bit que a compõe, como por exemplo, controlar uma linha de controle interna da UCP, controlar uma linha de barramento externo de controle, definir condição de desviro e endereço de desvio entre outras. Tem a vantagem de ser simples e direto possível, podendo controlar várias microoperações em paralelo, além de uma eficiente utilização do hardware. E possui a desvantagem de maior ocupação de espaço de memória de controle em relação à microinstrução vertical.

As microinstruções verticais se caracterizam por possuir um decodificador extra para identificar quais as linhas que serão efetivamente ativadas. Sua principal vantagem é reduzir o custo da Unidade de controle em função do menor tamanho da instrução, o que poderá ser necessário uma maior quantidade de instruções. Tem como principal desvantagem a redução do tempo devido à necessidade da decodificação dos campos de cada microinstrução.

4. (2,0) Um computador possui uma capacidade máxima de memória principal com 4 Giga células, cada uma capaz de armazenar uma palavra de 32 bits.

a) Qual é o maior endereço em decimal desta memória ?

N = 4G células => N = 2^{32}
Maior endereço = N - 1 = 2^{32} – 1 = 4.294.967.295

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

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 $2^e = N$*
$2^e = N$ => $2^e = 2^{32}$ => e = 32, portanto,
barramento de endereços = 32 bits

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

Tamanho do REM = tamanho do barramento de endereço. ***REM = 32 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?

O total de bits da memória será igual a T
Tamanho da memória (T) = N x M = 4G células x 32bits = 2^{32} x 2^5bits = ***2^{37}*** **bits**
= 137.438.953.472 bits

5. (2,0) Considere uma máquina que possa endereçar 1 Giga bytes de memória física, sendo que cada endereço referencia uma célula de 4 bytes. Ela possui uma memória cache que pode armazenar 1 Mega blocos, sendo um bloco por linha e cada bloco possui 2 células. Mostre o formato da memória cache, indicando os campos necessários (tag, bloco) e o número de bits para cada campo, o formato de um endereço da memória principal, indicando os bits que referenciam os campos da cache, e a capacidade em bits que a memória cache deve possuir (pode deixar a conta indicada) para os seguintes mapeamentos:

a) Mapeamento direto.

Memória Principal
⇒ *Tamanho da memória (em bytes) = 1Gbytes, como 1 célula referencia a 4 bytes, temos então, N = 256M células*

Será organizada em blocos de 2 células, temos cada bloco = 2 células, K = 2

⇒ *Sendo N o tamanho endereçável da memória, e K que a quantidade de células por blocos, N = 256 M células e K = 2 células / blocos, o total de blocos da MP (B) será: B = N / K => B = 256 M células / 2 células/bloco => B = 128 M blocos*

Memória Cache
OBS: O K (quantidade de células/bloco) tem de ser igual a MP.

⇒ *Tamanho da memória cache (em blocos ou linhas) => Q = 1M blocos*

⇒ *Tamanho da memória cache em células = Q x K = 1M blocos x 2células/bloco = 2M células*
Cada célula possui 4 bytes, então, cache possui 2M células x 4bytes, que totaliza 8M bytes

Endereço da MP: Para endereçarmos toda a MP precisamos da seguinte quantidade de bits (E) sendo $N = 2^E$ => N = 256 M células => $N = 2^{28}$ => E = 28 bits

Composição do endereço em função da memória cache
=> tag = B / Q = 128M / 1M = 128 = 2^7 = 7 bits
=> nº da linha: Q = 1M linhas ou quadros (máximo) => 2^{20} => 20bits
=> células por bloco: 2 células por bloco = 2^1 => 1 bit

28bits

Tag *7 bits*	*No. Linha* *20 bits*	*Célula no bloco* *1 bit*

b) Mapeamento totalmente associativo.

Memória Principal
=> N = 256 M células
=> K = 2 células/bloco
=> B = 128 M blocos

Memória Cache
OBS: O K (quantidade de células/bloco) tem de ser igual a MP.
=> Q = 1 M blocos
=> Tamanho da memória cache = 8Mbytes

Endereço da MP = 28 bits

Composição do endereço em função da memória cache
=> tag = B = 128 M blocos = 2^{27} => tag = 27bits
=> células por bloco: 2 células por bloco = 2^1 => 1 bit

28bits

Tag *27bits*	*Célula no bloco* *1 bit*

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

Memória Principal
=> N = 256 M células
=> K = 2 células/bloco
=> B = 128 M blocos

Memória Cache
OBS: O K (quantidade de células/bloco) tem de ser igual a MP.
=> Q = 1M blocos

=> Tamanho da memória cache = 8Mbytes
=> 1 conjunto = 2 linhas (ou quadros) =>
Total de conjuntos => C = 1M blocos / 2 => ***C = 512K conjuntos***

Endereço da MP = 28 bits

Composição do endereço em função da memória cache
=> tag = B / C = 128 M / 512K = 256 = 2^8 => tag = 8 bits
=> nº do Conjunto: Q = 512K conjuntos => 2^{19} => 19 bits
=> células por bloco: 2 células por bloco = 2^1 => 1 bit

28 bits

Tag *8 bits*	*No. Conjunto* *19 bits*	*Célula no bloco* *1 bit*