AD1 - Organização de Computadores 2010.1
Data de entrega 27/03/2010

1. **(1,0) Descreva passo a passo as operações de leitura da memória e de escrita na memória, indicando como os registradores RDM e REM são utilizados e como a unidade de controle gera os sinais necessários.**

   Passos de uma operação de escrita
   1) (REM) ← (outro registrador)
   1.1)O endereço é colocado no barramento de endereços
   2) (RDM) ← (outro registrador)
   2.1)O dado é colocado no barramento de dados
   3) Sinal de escrita é colocado no barramento de controle
   4) (MP(REM)) ← (RDM)

   Passos de uma operação de leitura
   1) (REM) ← (outro registrador)
   1.1) O endereço é colocado no barramento de endereços
   2) Sinal de leitura é colocado no barramento de controle
   2.1) Decodificação do endereço e localização da célula na memória
   3) (RDM) ← (MP(REM)) pelo barramento de dados
   4) (outro registrador) ← (RDM)

2. **(1,0) Faça uma pesquisa de um processador qualquer que possua múltiplos núcleos de processamento (processadores multicore). Considerando esse processador, descreva sua hierarquia de memória e como os núcleos acessam as memórias caches e a memória RAM. Indique os registradores, barramentos e sinas de controle utilizados.**

OBS: Estão indicadas nos tópicos, as respectivas fontes onde foram retirados os textos.

Abordagem inicial de alguns termos da arquitetura multicore: (fonte www.unversopc.net)

- ***Dual Core*** *é todo processador que possui dois núcleos de processamento. Alguns exemplos de processadores* ***Dual Core:*** *Core 2 Duo, Athlon X2, Pentium D, Pentium Dual Core*
- ***Core 2 Duo*** *é o nome de uma linha de processadores Dual Core da Intel. Existem também os processadores Core 2 Quad (com quatro núcleos) e os modelos Core 2 Extreme (Alta performance)*
- ***Quad Core*** *é todo processador com quatro núcleos, como por exemplo, o Core 2 Quad e o Phenom X4.*

Vale salientar que apenas substituindo um processador single core por um processador Dual Core o desempenho do PC não dobra, porque a velocidade do computador depende de outros fatores como placa de vídeo, tempos de acesso às memórias, tecnologia do disco magnético, etc.

Descrição de alguns processadores muti-core da Intel e a organização das caches (fonte: coluna do Laércio Vasconcelos no site www.forumpcs.com.br)

O Pentium D possui dual core (dois processadores reais) ao invés da tecnologia HT (dois processadores virtuais). Já o Pentium Extreme Edition tem dois núcleos, cada um deles operando com HT. Este processador é visto então pelos programas como quatro processadores.
Processadores Pentium D e Pentium Extreme Edition eram na verdade formados por dois núcleos de Pentium 4, interligados e encapsulados juntos (figura 1). Esse núcleo usava a arquitetura chamada Intel Netburst. Apesar dos clocks serem elevados, essa arquitetura era menos eficiente que as utilizadas em outros processadores contemporâneos, como o Athlon e o Pentium M (plataforma Centrino, para notebooks). Por isso um Pentium 4 precisava operar com clock muito elevado para ter bom desempenho. Um Pentium 4 de 3.2 GHz tinha um desempenho próximo ao de um Athlon 64 3200, que opera com apenas 2.0 GHz.

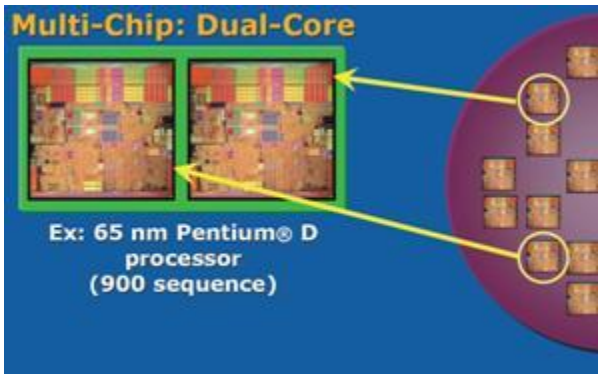


Figura 1: Pastilha do Pentium D
(retirada da coluna do Laércio Vasconcelos
do site www.forumpcs.com.br)

Em 2006 a Intel criou a nova arquitetura Core, baseada no núcleo do Pentium M, para substituir a arquitetura Netburst, que já completava seis anos. O Pentium D e o Pentium Extreme Edition usavam núcleos Netburst. Os novos processadores Core 2 Duo, Core 2 Quad e Core 2 Extreme usam a nova arquitetura Core, muito mais eficiente que a NetBurst.

Organização das caches:Cada chip independente já é uma dupla de núcleos, interligados a uma cache L2 compartilhada. Cada núcleo usa a princípio a metade da cache L2, mas um núcleo pode usar parte da cache do outro núcleo. O resultado é um melhor aproveitamento da cache, e maior desempenho.
A cache L1 possui 64 KB (32 KB de dados + 32 KB de instruções) por núcleo para as arquiteturas multicore. A cache de memória L2 do Pentium Dual Core é de 1 MB, compartilhada entre os núcleos (a Intel chama esta implementação de cache L2 compartilhada de Smart Cache, ou cache inteligente) e ele trabalha externamente a 800 MHz (200 MHz transferindo quatro dados por pulso de clock). A título de comparação, os processadores Core 2 Duo possuem pelo menos 2 MB de memória cache L2 sendo que diversos modelos possuem 4 MB. Apesar de existirem modelos com barramento externo de 800 MHz, a maioria usa barramento de 1.066 MHz, como modelos mais recentes rodando externamente a 1.333 MHz.

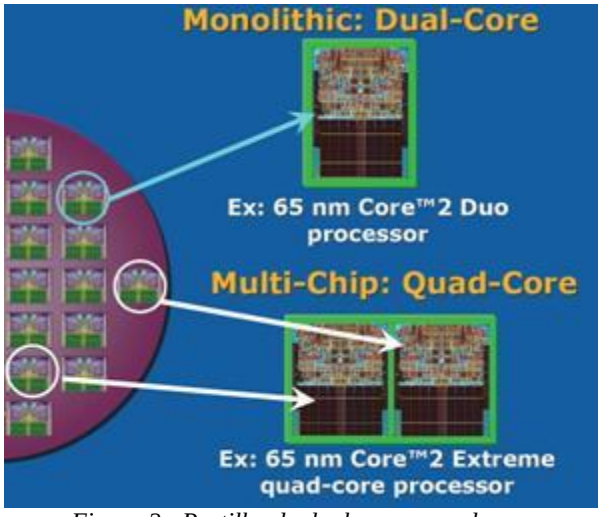


Figura 2: Pastilha do dual-core e quad-core
(retirada da coluna do Laércio Vasconcelos
do site www.forumpcs.com.br)

Cada pastilha de silício do Core 2 Duo integra dois núcleos. Processadores de quatro núcleos (Core 2 Quad e Core 2 Extreme) são formados por duas dessas pastilhas integradas no mesmo chip, formando quatro núcleos

A Intel disponibiliza para o mercado de servidores e estações de trabalho a série Xeon (fonte: fonte: www.clubedohardware.com.br)

Xeon com 2 núcleos: Os processadores Xeon 31xx e 52xx, assim como os modelos 30xx, 51xx e 72xx, são baseados na microarquitetura Core. A principal diferença entre esses modelos é a tecnologia de fabricação. Enquanto os modelos 30xx, 51xx e 72xx usam o processo de fabricação de 65 nm, as séries 31xx e 52xx usam o novo processo de 45 nm. Possui cache L1 dividida, sendo 32 KB para dados e 32 KB para instruções por núcleo. A cache L2 de 6 MB é compartilhada entre os núcleos.
Xeon com 4 núcleos: Os processadores Xeon Séries 33xx e 54xx, assim como os modelos 32xx, 53xx e 73xx, são baseados na microarquitetura Core, a mesma usada pelos processadores Core 2 Duo. Enquanto os modelos 32xx, 53xx e 73xx usam o processo de fabricação de 65 nm, as séries 33xx e 54xx usam o novo processo de 45 nm. Os quatro núcleos dos processadores Xeon 32xx, 53xx e 73xx bem como as séries 33xx e 54xx são obtidos a partir de duas pastilhas de dois núcleos cada, assim como ocorre com os modelos descritos anteriormente. Com isso, a cache L2 desses processadores não é compartilhada entre todos os seus núcleos: os núcleos 1 e 2 compartilham uma mesma cache L2, enquanto que os núcleos 3 e 4 compartilham uma outra cache L2. O valor divulgado é o valor total (soma das duas caches). A cache L1 é dividida, sendo 32 KB para dados e 32 KB para instruções por núcleo e a cache L2 de 4 MB, 6 MB, 8 MB ou 12MB, dependendo do modelo, dividida em dois

Descrição de processadores muti-core da AMD e organização das caches (fonte: coluna do Laércio Vasconcelos no site www.forumpcs.com.br)

Como representante da AMD na arquitetura multi-core, tem-se o Athlon 64 X2. Praticamente na mesma época do lançamento do Pentium D e do Pentium Extreme Edition (segundo trimestre de 2005), a AMD também lançou seus processadores duais. São novos modelos do processador Opteron, para servidores, e o Athlon 64 X2, para uso em desktops, usando placas com Socket 939. Posteriormente foram lançados novos modelos com o Socket AM2, com suporte a memórias DDR2. Com seus dois núcleos, seu desempenho tende a ser de 50% a 100% maior, dependendo da aplicação. Os primeiros modelos foram produzidos com a tecnologia de fabricação de 90 nm, depois foram lançados novos modelos com 65 nm.
Organização das caches L1 e L2: Todos os modelos de Athlon 64 X2 possuem duas caches L1 de 128 kB, sendo uma para cada núcleo. Cada núcleo também possui sua própria cache L2, mas o tamanho varia de acordo com o modelo, em torno de 1MB.
Os processadores duais da AMD têm uma característica superior aos processadores Intel de primeira geração (Pentium D e Pentium Extreme Edition). A comunicação entre os dois núcleos é interna. Com isso existem duas grandes vantagens:

1) *Maior desempenho na comunicação entre os processos em execução nos dois núcleos.*
2) *Do ponto de vista externo, o processador é visto pela placa mãe como um processador não dual. Por isso o Athlon 64 X2 não requer chipsets especiais.*

Controlador de memória e Operações de leitura e escrita (fonte: coluna do Laércio Vasconcelos no site www.forumpcs.com.br em parte)

*Em um PC atual o Processador é o componente mais rápido, seguido pela cache e pela memória RAM. A memória RAM não faz nada sozinha, ela é apenas um depósito de dados. Quem controla a gravação e leitura dos dados, administra os endereços disponíveis, e mantém tudo organizado é o controlador de memória, que faz parte do **chipset** da placa mãe. O processador não se comunica diretamente com a memória RAM, mas sim com o controlador de memória, que se encarrega de todo gerenciamento da comunicação.*
A cache L2 por sua vez também possui seu controlador. Este controlador está embutido no próprio processador, e é desenvolvido de modo a guardar apenas os dados que possuem maior possibilidade de serem usados mais tarde pelo processador, e a tentar "adivinhar", com base nas instruções que o processador está executando, quais serão os próximos dados que ele irá precisar. A cache L1 por sua vez tem um funcionamento semelhante com seu próprio controlador, sempre embutido no processador. A vantagem da cache L1 sobre a L2 é ser bem mais rápida.

Sempre que a UCP vai buscar uma nova instrução (ou dado), ela primeiro acessa a memória cache

- *Se a instrução estiver na cache, chama-se de acerto (hit). Ela é transferida em alta velocidade.*
- *Se a instrução não estiver na cache, chama-se de falta (miss).*
- *O bloco de instruções a que a instrução pertence é transferida da MP para a cache, considerando o princípio da localidade.*

Para uma operação de escrita na MP, a UCP envia o endereço através do registrador REM (e consequentemente, para o barramento de endereços), o dado pelo RDM (e consequentemente, para o barramento de dados), e um sinal de escrita (WR). Com este sinal, a memória armazena o dado fornecido na posição contida no barramento de endereços.
Para uma operação de leitura, a UCP envia o endereço através do registrador REM (e consequentemente, para o barramento de endereços) e o sinal de Leitura (RD). Com este sinal, a memória busca na posição de memória,, fornecida no barramento de endereços, a informação que será colocada no barramento de dados (e consequentemente, para o RDM), retornando, assim, à UCP.
No exemplo apresentado na figura 3, a organização da memória é do tipo ***matriz-coluna****, presente nas arquiteturas tradicionais, onde os endereços fornecidos para memória são divididos em 2 (coluna e linha), o barramento para a memória corresponderá a metade do barramento de endereços do processador. Quando é colocado o endereço correspondente à coluna, o sinal de CAS(Column Access Memory) é ativado (nível baixo), e quando o endereço corresponde às linhas, o sinal de RAS(Row Access memory) é ativado (nível baixo). A operação de leitura é ativada no nível baixo do sinal de OE (RD) e o de escrita, no nível baixo de W.*

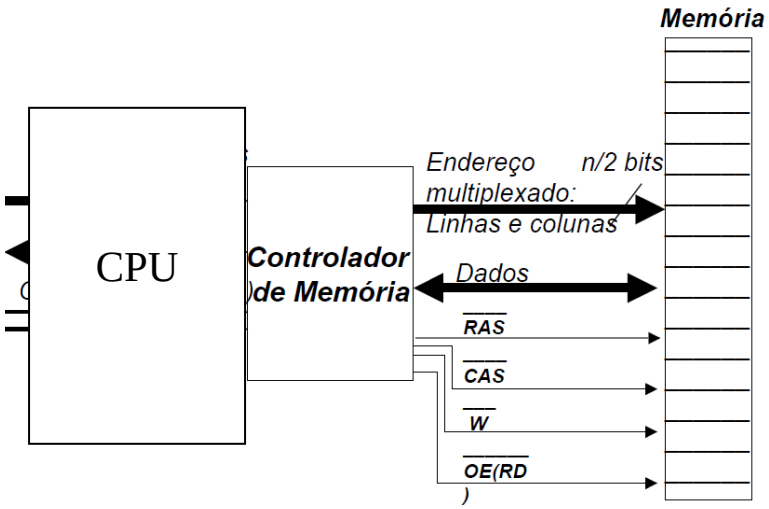


Figura 3: Acesso à memória principal

O controlador de memória dos computadores atuais (multicore) utilizam o tipo de seleção linear, não necessitando enviar endereços divididos em coluna/linha. O controlador de memória de um sistema Intel típico fica situado no chipset da placa-mãe. Todos os dados a serem processados pela CPU, e os resultados desse processamento passam por esse controlador de memória, sendo que os dados só podem fluir em um sentido de cada vez, similar a um semáforo, que controla o trânsito em uma estrada onde apenas um carro pode circular de cada vez.

Em ambientes multi-processados, os processadores Intel tem que dividir sua largura de banda entre si para cada processador adicionado ao sistema, pois teremos vários processadores querendo se comunicar com o mesmo controlador de memória, enquanto os processadores AMD64 por ter o controlador embutido, multiplicam a largura de banda para cada processador adicionado.

A Intel, ao desenvolver o DualCore, simplesmente colocou dois núcleos Prescott em uma única pastilha de silício, sem nenhum mecanismo especial para a comunicação entre eles. Tal comunicação é feita através do FSB (Front Side Bus) externo. Dessa forma a comunicação entre os dois núcleos é lenta, já que além de se comunicarem através do FSB externo os núcleos compartilham a banda da memória através do próprio barramento externo, o que aumenta o tempo de acesso à memória RAM.

Os mais novos processadores da AMD possuem controlador de memória dentro do próprio processador e não do chipset como acontece em outros processadores. Assim, a comunicação entre o processador e os módulos de memória é feita através de um barramento dedicado, enquanto que a comunicação entre o processador e o chipset é feita através de um barramento independente, chamado HyperTransport. Logo, o caminho de dados entre o processador e o controlador de memória utiliza o clock interno do processador em vez do clock externo, como acontece com os processadores da Intel. A Intel busca corrigir este problema com a recente série Intel Core IX, incluindo o controlador de memória dentro do processador.

3. (1,0) Um computador possui uma capacidade máxima de memória principal com 4 Gbytes células, cada uma capaz de armazenar uma palavra de 8 bits.

OBS: Nesta questão, a capacidade máxima da memória principal é de 4 G células (valor de N), cada uma capaz de armazenar uma palavra de 8 bits (1 byte, valor de M), portanto, capacidade total é de 4 Gbytes (valor de T). Apesar de ter saído no título "4Gbytes células", esta dupla especificação não influenciará nas respostas, que serão apresentadas nos 2 entendimentos.

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

Partindo de T = 4Gbytes

T = N x M => 4Gbytes = N x 1byte (8bits) => N = 4Gcélulas = 2^{32}

Partindo de *N = 4G células = 2^{32}*

Então,

Último endereço = N - 1 = 2^{32} = 4.294.967.296 - 1 = ***4.294.967.295***

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

Barramento de endereços = E

N = 2^{E} = 2^{32}, portanto E = 32, Barramento de endereços = ***32 bits***

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

O REM terá que ter o tamanho do barramento de endereços = ***32 bits***

RDM = tamanho da palavra = ***8 bits***

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

T = 4Gbytes = 4 G x 8bits = 32 G bits

ou

T = N x M => T = 4G células x 8 bits = 32 G bits

4. (1,0) Considere uma máquina que possa endereçar 1Gbytes de memória física, utilizando endereço referenciando byte, e que tenha a sua memória organizada em blocos de 4 Kbytes. Ela possui uma memória cache que pode armazenar 2K blocos, sendo um bloco por linha. Mostre o formato da memória cache, indicando os campos necessários (válido, 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:

a) Mapeamento direto.

Memória Principal

⇒ *Tamanho da memória (em bytes) = 1Gbytes, como 1 célula referencia a 1 byte, temos N = 1 G 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 que é a quantidade de células por blocos temos:*
N = 1 G células e K = 4K células / bloco o total de blocos da MP (B) será:
Total de blocos: B = N / K => B = 1 G células / 4 K células/bloco => B = 256 K blocos

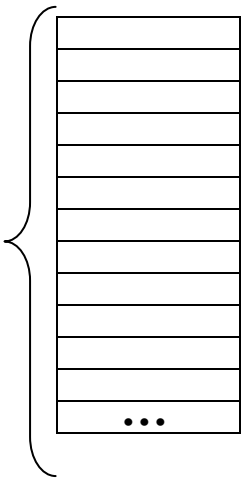
Memória Cache

OBS: A quantidade de células por bloco tem de ser igual a MP.

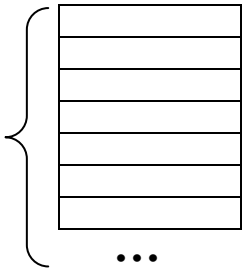
⇒ *Tamanho da memória cache em blocos = 2 K blocos*

⇒ *Tamanho da memória cache em células = 2k blocos x 4k célula/bloco = 8M células*

1 G células: (N)
256K blocos:(B)



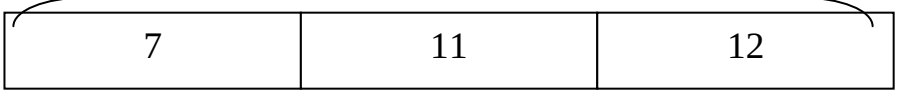
2K blocos



Cada célula corresponde a 1 byte (8bits)
Cada bloco (MP) e linha (cache) corresponde a 4K células (K)

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

Tamanho do endereço da MP = 30 bits



Corresponde a tag:
tag = B / Q = 256K / 2K = 128 = 2^7

Corresponde ao nº da linha:
Q = 2K linhas ou quadros (máximo) => 2^{11}

Corresponde ao endereço da célula no bloco:
4 K células por bloco = 2^{12}

b) Mapeamento totalmente associativo.

Memória Principal

=> N = 1G células

=> K = 4K células por bloco

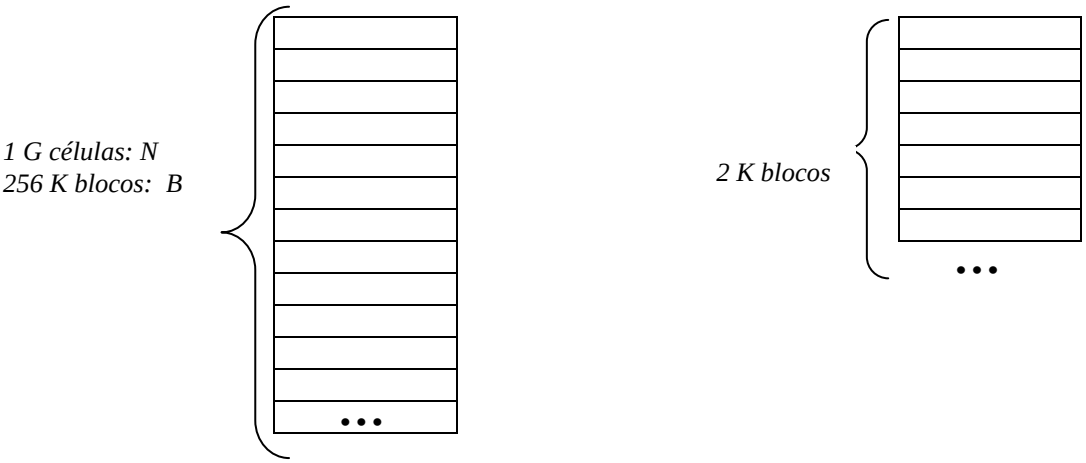
=> B = 256 K blocos

Memória Cache

OBS: A quantidade de células por bloco tem de ser igual a MP.

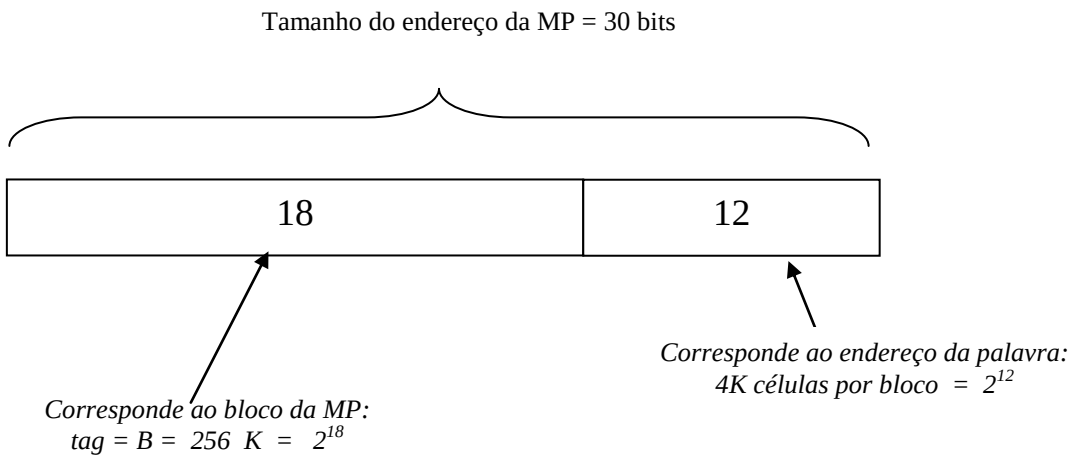
=> Q = 2K blocos

=> Tamanho da memória cache = 8 M células



Para endereçarmos toda a MP precisamos da seguinte quantidade de bits: E = 30 bits

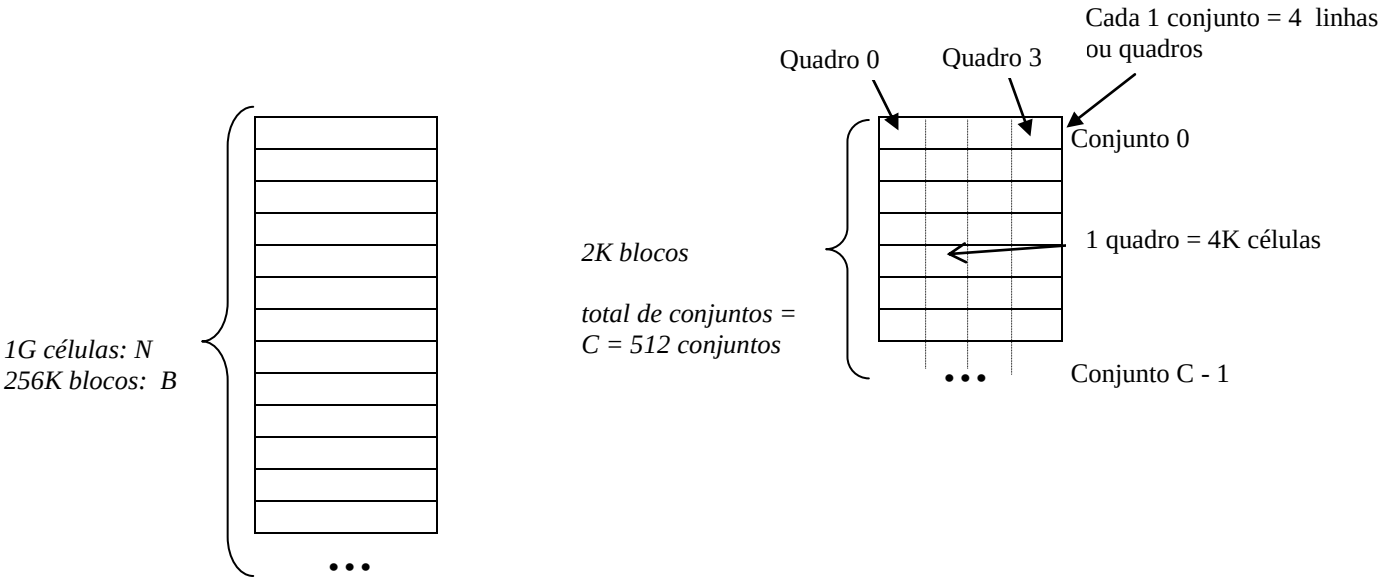
Como o bloco pode ser alocado em qualquer posição da memória cache a tag indicará qual dos blocos da MP está alocado naquela posição da memória cachê



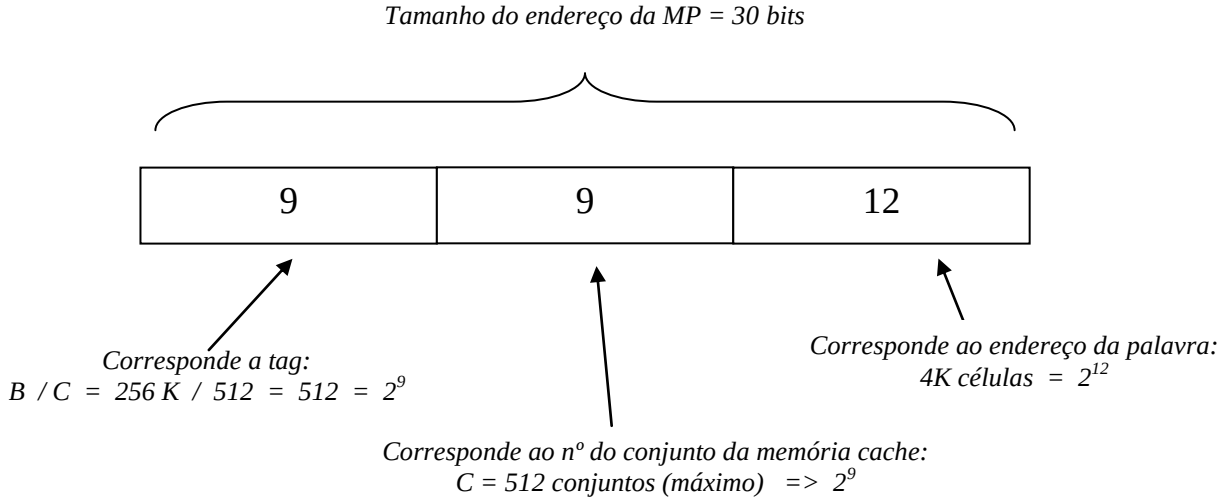
c) Mapeamento associativo por conjunto, onde cada conjunto possui quatro linhas, cada uma de um bloco.

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

Memória Cache
OBS: A quantidade de células/bloco tem de ser igual a MP.
=> Q = 2K blocos
=> Tamanho da memória cache = 8M células
=> 1 conjunto = 4 linhas (ou quadros) =>
Total de conjuntos => C = 2K células / 4 => ***C = 512 conjuntos***



Para endereçarmos toda a MP precisamos da seguinte quantidade de bits: E = 30 bits



5. (1,0) Considere a máquina apresentada na aula 4. Descreva detalhadamente (do mesmo modo que é apresentado na aula 4) como é realizada a execução das seguintes instruções:

a) LDA 300

1. *RI ← Instrução lida*
2. *CI ← CI + 1*
3. *Decodificação do código de operação*
4. *Busca do operando na memória*
   - i. *A UC emite sinais para que o valor do campo operando = 300 seja transferido para o REM*
   - ii. *A UC emite sinais para que o valor contido no REM seja transferido para o barram*ento de endereços
   - iii. *A UC ativa a linha READ do barramento de controle*
   - iv. *Conteúdo da posição da memória, conforme endereço contido no barramento de endereços (300), é transferido através do barramento de dados para o RDM*
5. *O conteúdo do RDM é transferido para o registrador acumulador (ACC ← RDM)*

b) JP 400

1. *A CPU verifica se o valor contido no ACC é positivo (maior que zero)*
   - i. *Caso seja verdadeira a verificação:*
     -CI ← Op. Sendo Op = 400.
     -Será executada a instrução do endereço 400.
   - ii. *Caso seja falsa a verificação:*
     -CI ← CI + 1.
     -Será executada a instrução do endereço seguinte

6. (1,0) Escreva um programa que utilize as instruções de linguagem de montagem apresentadas na aula 4 para executar o seguinte procedimento. O conteúdo da memória cujo endereço é 50 é lido e verifica-se se o seu valor o é maior que 0. Caso seu valor seja maior que 0, o conteúdo de memória cujo endereço é 60 é subtraído do conteúdo de memória cujo endereço é 50 e o resultado é armazenado no endereço 50. Caso contrário, o conteúdo de memória cujo endereço é 50 é multiplicado por 2 e o resultado é armazenado no endereço 50.
Além de apresentar seu programa escrito em linguagem de montagem, apresente também o programa traduzido para linguagem de máquina.

OBS: Considerando endereços em hexadecimal ou em decimal, com instruções = tamanho da célula = 12 bits e códigos de operação com 4 bits. Considerou-se também que o programa foi armazenado a partir do endereço 10. Soluções que considerem outro endereço inicial de armazenamento devem ser consideradas. A única modificação será na segunda instrução (JP 15) que deve mudar para (JP end.inicial +5)

Endereço	***Instrução***	***Descrição***	***Linguagem Máquina (bin / hexa)***	
Em dec. Ou hex			*Considerando Endereços em hexa*	*Considerando Endereços em decimal*
10	*LDA 50*	*ACC ← (50)*	*(0001 01010000 / 150)*	*(0001 00110010 / 132)*
11	*JP 15*	*se ACC>0, CI ← 15*	*(0110 00010101 / 615)*	*(0110 00001111 / 60F)*
12	*ADD 50*	*ACC ← ACC + (50)*	*(0011 01010000 / 350)*	*(0011 00110010 / 332)*
13	*STR 50*	*(50) ← ACC*	*(0010 01010000 / 250)*	*(0010 00110010 / 232)*
14	*HLT*	*Encerra*	*(0000 00000000 / 000)*	*(0000 00000000 / 000)*
15	*SUB 60*	*ACC ← ACC - (60)*	*(0100 01100000 / 460)*	*(0100 00111100 / 43C)*
16	*STR 50*	*(50) ← ACC*	*(0010 01010000 / 250)*	*(0010 00110010 / 232)*
17	*HLT*	*Encerra*	*(0000 00000000 / 000)*	*(0000 00000000 / 000)*

7. (2,0) Considere uma máquina com arquitetura semelhante àquela apresentada em aula. Pode-se endereçar no máximo 128 M células de memória, sendo que cada célula tem tamanho igual a 16 bits. Em cada acesso á memória, obtém-se o conteúdo de uma célula. Todas as instruções desta máquina possuem dois campos: o primeiro indica o código de operação e o segundo indica endereço de célula de memória onde se encontra o operando. Esta máquina possui 30 códigos de operação diferentes.

a) Calcule a capacidade mínima de endereçamento em bits do REM, considerando que os bits armazenados no REM são utilizados para endereçar uma célula de memória.

REM = E = tamanho em bits necessários para acessar toda a memória (N)
$N = 2^E$ = 128M células = 2^{27} células => E = 27 => REM = 27 bits

b) Calcule o número de bits que devem poder ser transmitidos no barramento de endereços em cada acesso à memória.

Barramento de endereços = REM = 27 bits

c) Calcule o tamanho do RI (Registrador de Instruções).

Tamanho mínimo de RI = tamanho da instrução
Tamanho da instrução = código de operação + 1 operando
Tamanho da instrução = (tamanho p/ endereçar 30 instruções)
+ 1 x (endereço de memória)
Tamanho da instrução = 5 bits + 1 x 27 bits = 32 bits
Tamanho mínimo de RI = 32bits

d) Calcule o número de células que uma instrução ocupa.

Tamanho mínimo de 1 instrução = 32 bits
Tamanho de 1 célula = 16 bits
Serão necessárias pelo menos 2 células para uma instrução

e) Calcule a capacidade máxima de armazenamento da memória deste sistema em bits.

T = N x M => T = 128 Mcélulas x 16 bits/célula => T = 2048 Mbits

f) Calcule a capacidade mínima de endereçamento em bits do CI (Contador de Instrução), considerando que os bits armazenados no CI são utilizados para endereçar a primeira célula de uma instrução armazenada na memória.

CI = tamanho necessário para armazenar o endereço de uma célula da memória = 27 bits

8. (1,0) Considere uma máquina cujo relógio possui uma freqüência de 3 GHZ e um programa P1 no qual são executadas 50 instruções desta máquina e um programa P2 no qual são executadas 1000 instruções.

a) Calcule o tempo para executar os programas P1 e P2, considerando que cada instrução é executada em 6 ciclos de relógio e a execução de uma instrução só se inicia quando a execução da instrução anterior é finalizada.

Tempo de um ciclo de relógio = 1/3.000.000.000 = 1/3 x 10^{-9} seg ou 1/3 ns (nanosegundos)
Tempo de execução de 1 instrução = 6 ciclos de relógio = 6 x 1/3 ns = 2ns

Para o programa P1
*100 instruções executadas seqüencialmente = 50 x 2ns = **100ns***

Para o programa P2
*1000 instruções executadas seqüencialmente = 1000 x 2ns = **2000ns***

b) Uma nova implementação dessa máquina utiliza um pipeline de 5 estágios, todos de duração igual a 2 ciclos de relógio. Calcule o tempo para executar os programas P1 e P2, considerando que não existem conflitos de qualquer tipo.

Tempo para um estágio = 2 ciclos de relógio = 2 x 1/3 ns = 2/3 ns
Para execução da 1ª instrução = 5 estágios x 2/3ns = 10/3 ns
Para execução das instruções posteriores = tempo de 1 estágio devido ao pipeline = 2/3 ns

Para o programa P1
Tempo total para execução das 50 instruções = 10/3ns + 49 x 2/3ns = 108/3 ns ***= 36 ns***

Para o programa P2
Tempo total para execução das 1000 instruções = 10/3ns + 999 x 2/3ns = 2008/3 ns = ***669,333 ns***

c) Calcule, para o programa P1 e para o programa P2, a relação entre os tempos de execução obtidos no item a com os tempos obtidos no item b.

Para o programa P1
Relação = 100ns / 36ns = 2,78, isto é, na execução do programa P1, a arquitetura sem pipeline obteve um tempo de execução 2,78 vezes maior do que a arquitetura com pipeline

Para o programa P2
Relação = 2000ns / 699,333ns = 2,86, isto é, na execução do programa P2, a arquitetura sem pipeline obteve um tempo de execução 2,86 vezes maior do que a arquitetura com pipeline

9. (1,0) Faça uma pesquisa e indique um microprocessador comercial, cuja unidade de controle possa ser caracterizada como sendo por hardware e um outro, cuja unidade de controle possa ser caracterizada por microprogramada. Explique detalhadamente as suas indicações.

Unidades microprogramadas

A unidade de controle microprogramada é utilizada para se desenvolver a implementação de um conjunto de instruções que apresenta muita complexidade para ser implementado somente em hardware. Implementado-se a unidade de controle por microprograma torna mais fácil seu projeto, implementação e verificação de erros.

A Unidade de Controle microprogramada é composta por:
- Memória de Controle: memória que contem as microinstruções
- Contador de microprograma: registrador que armazena a localização da próxima instrução a ser executada
- Seqüenciador: componente que controla a sequencia de execução das microinstruções. Informa o local da próxima microinstrução que deve ser executada e armazenada no contador de microprograma.

As unidades de controle microprogramadas estão presentes nos processadores de arquitetura CISC (Complex Set Instruction Computer). Nesta arquitetura, o conjunto de instruções é extenso e as instruções possuem formatos diversos, o que torna mais complexa a implementação da unidade de controle. Esta arquitetura tem como exemplo: processadores da série Intel (Pentium, Xeon, Core 2 Duo, Dual Core, Quad Core, e demais variantes), e seus compatíveis da AMD.

Unidades de controle por hardware

O desenvolvimento de uma unidade de controle implementada por hardware consiste essencialmente em projetar circuitos combinatórios. Os sinais lógicos de entrada na unidade devem ser transformados em um conjunto lógico

de sinais que controlam a execução da instrução. Para implementar a unidade, a partir dos sinais de entrada deriva-se uma expressão booleana para cada sinal de controle necessário para que a instrução seja executada de forma correta. O número de expressões booleanas pode ser muito grande, o que pode tornar difícil a implementação de um conjunto de instruções que possua um grande número de instruções. A vantagem da implementação por hardware sobre microprogramação está na velocidade de execução das instruções. Esta velocidade de execução pode compensar o baixo número de instruções presente no conjunto de instruções que utiliza este tipo de implementação da unidade de controle, o que viabiliza este tipo de arquitetura.

As unidades de controle por hardware estão presentes nos processadores das arquiteturas RISC (Reduced Set Instruction Computer). Nesta arquitetura, o conjunto de instruções é pequeno e as instruções possuem um único formato, o que torna mais simples a implementação da unidade de controle. Esta arquitetura tem como exemplos: a série Sparc da Sun e, também, os da serie Power PC da IBM