GABARITO AD1 - Organização de Computadores 2009.2

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 da UCP)
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 da UCP) <- (RDM)

2. (1,0) Um computador possui uma capacidade máxima de memória principal com 64K células, cada uma capaz de armazenar uma palavra de 8 bits.

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

N = 64K células = 2^{16}
*Último endereço = N - 1 = 65.536 -1 = **65.535***

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

Barramento de endereços = E
*$N = 2^E = 2^{16}$, portanto E = 16, Barramento de endereços = **16 bits***

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

*O REM terá que ter o tamanho do barramento de endereços = **16 bits***
CI terá o tamanho necessário para endereçar toda a memória = 16 bits
*RDM = tamanho da palavra = **8 bits***

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

*T = N x M = 64K células x 8 bits/célula = 512 **Kbits***

3. (1,0) Considere uma máquina que possa endereçar 1Gbyte 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 1M células, 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*

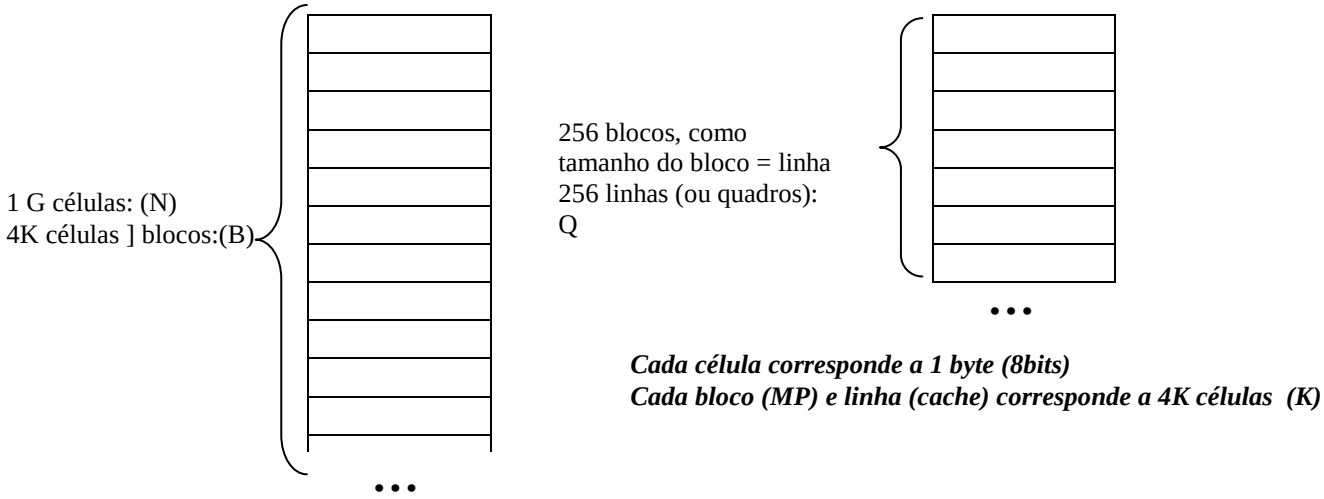
⇒ *Sendo N o tamanho endereçável da memória e K que é a quantidade de células por blocos temos:*
N = 1 G células e K = 4K células / blocos o total de blocos da MP (B) será:
Total de blocos: B = N / K => B = 1 G células / 4 K células/bloco => B = 256 K blocos

Memória Cache

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

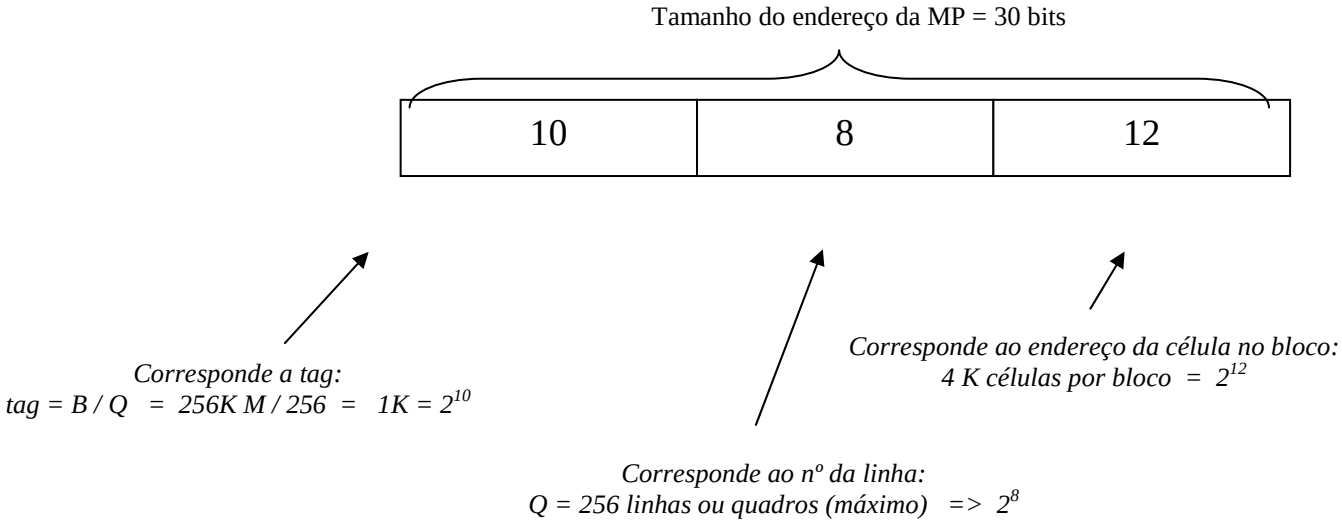
⇒ *Tamanho da memória cachê em células = 1 M células*

⇒ *Tamanho da memória cache (em blocos ou linhas) Tamanho da memória cachê em células / bloco =*
= 1M células / 4 K células = 256 blocos



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

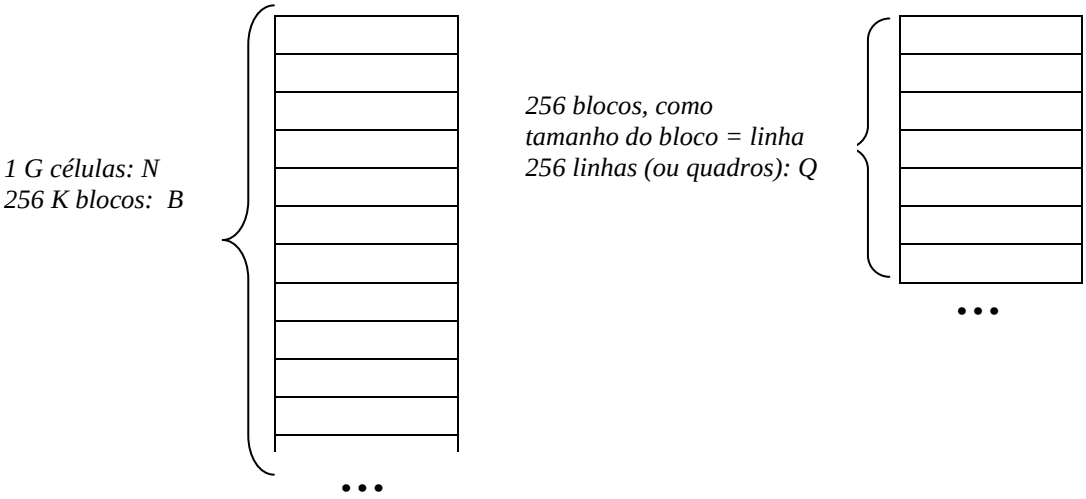
Para endereçarmos toda a MP precisamos da seguinte quantidade de bits (E)
sendo $N = 2^E$ => N = 1G células => $N = 2^{30}$ => E = 30 bits



b) Mapeamento totalmente associativo.

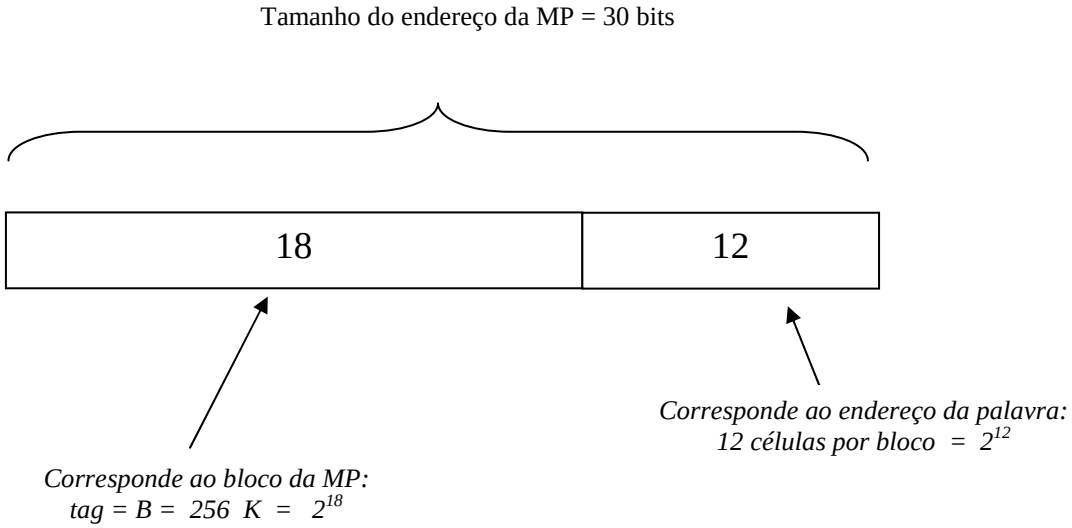
Memória Principal
=> N = 1G células
=> K = 4K células por bloco
=> B = 256 K blocos

Memória Cache
OBS: O K (quantidade de células/bloco) tem de ser igual a MP.
=> Q = 256 blocos
=> Tamanho da memória cache = 1 M células



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

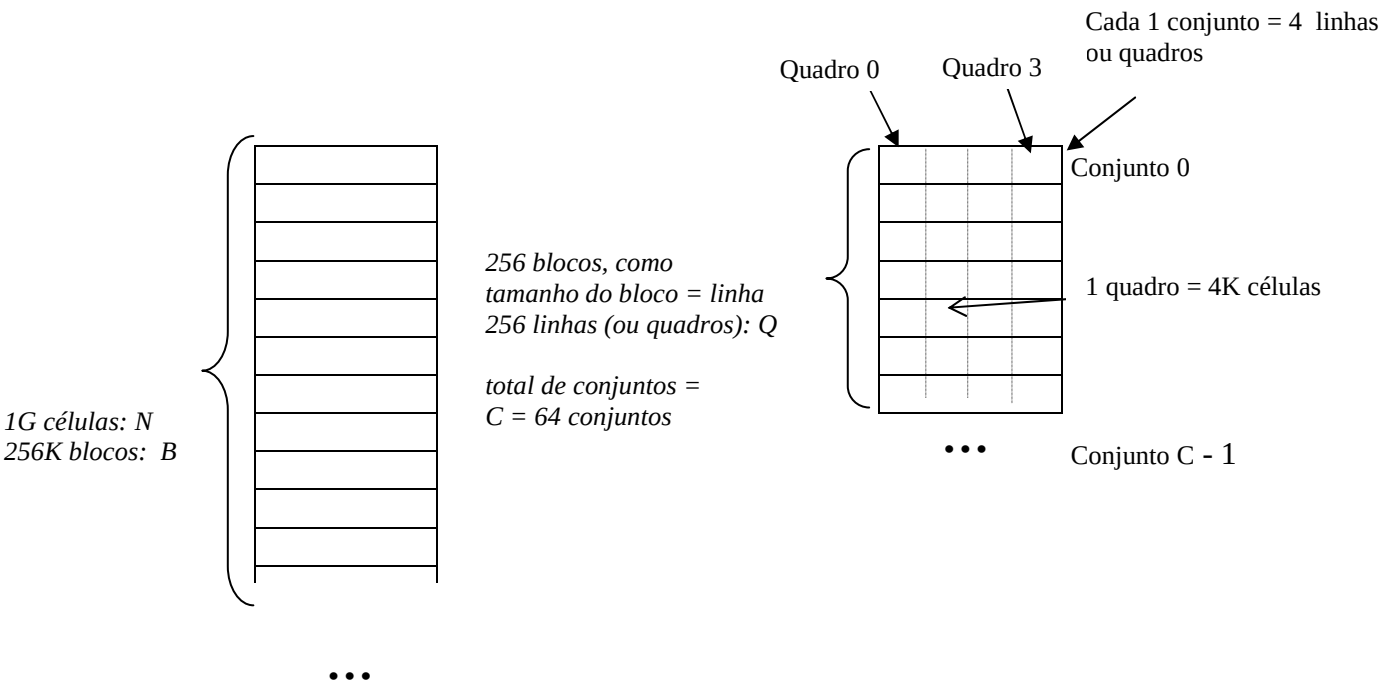
Como o bloco pode ser alocado em qualquer posição da memória cacha a tag indicará qual dos blocos da MP está alocado naquela posição da memória cachê



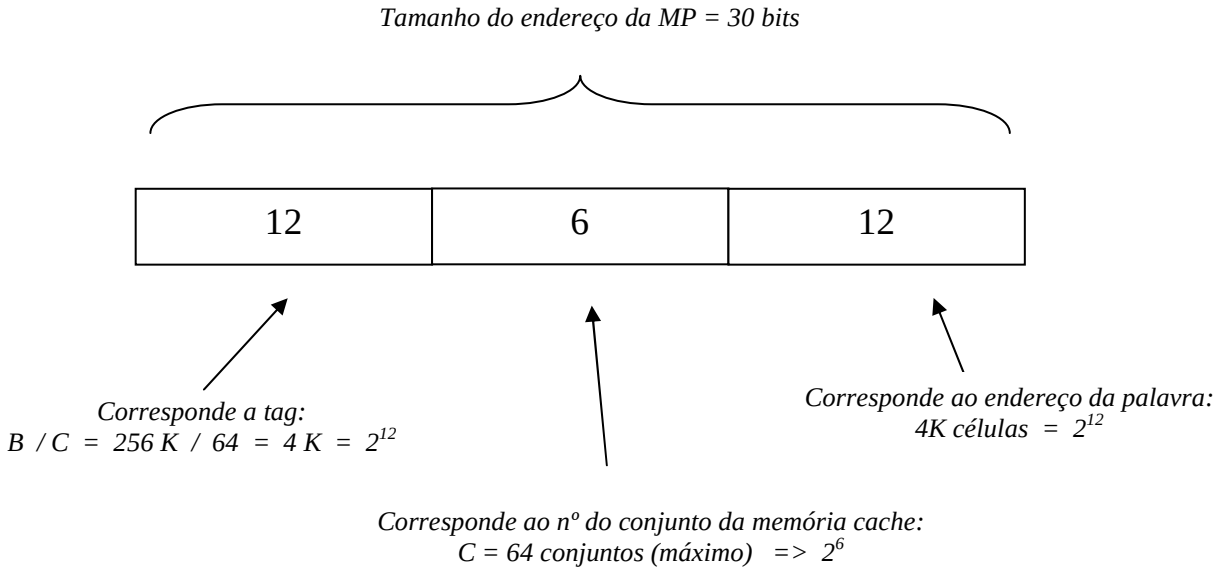
c) Mapeamento associativo por conjunto, onde cada conjunto possui quatro linhas, cada uma de um bloco.

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

Memória Cache
OBS: O K (quantidade de células/bloco) tem de ser igual a MP.
=> Q = 256 blocos
=> Tamanho da memória cache = 1M células
=> 1 conjunto = 4 linhas (ou quadros) =>
Total de conjuntos => C = 256 células / 4 => ***C = 64 conjuntos***



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



4. (1,0) Faça uma pesquisa e explique a organização hierárquica do subsistema de memória nos computadores atuais, dando detalhes em particular da hierarquia de memória dos processadores Intel Xeon.

Organização hierárquica do subsistema de memória em computadores atuais:

O subsistema de memória é interligado de forma bem estruturada e organizado hierarquicamente em uma pirâmide com os níveis descritos a seguir.

No topo da pirâmide teríamos os registradores, que são pequenas unidades de memória que armazenam dados na UCP. São dispositivos de maior velocidade com tempo de acesso em torno de 1 ciclo de memória, menor capacidade de armazenamento além de armazenar as informações por muito pouco tempo.

Em um nível abaixo teríamos a memória cache, cuja função é acelerar a velocidade de transferência das informações entre UCP e MP e, com isso, aumentar o desempenho do sistema. A UCP procura informações primeiro na Cache. Caso não as encontre, as mesmas são transferidas da MP para a Cache. A cache possui tempo de acesso menor que a da Memória principal, porém com capacidade inferior a esta, mas superior ao dos registradores e o suficiente para armazenar uma apreciável quantidade de informações, sendo o tempo de permanência do dado menor do que o tempo de duração do programa a que pertence.

Abaixo da memória cache teríamos a memória básica de um sistema de computação, que é a memória principal. Dispositivo onde o programa (e seus dados) que vai ser executado é armazenado para que a UCP busque instrução por instrução para executá-las. A MP são mais lentas que a cache e mais rápidas que a memória secundária, possui capacidade bem superior ao da cache e os dados ou instruções permanecem na MP enquanto durar a execução do programa.

Finalmente, na base da pirâmide teríamos a memória secundária, memória auxiliar ou memória de massa, que fornece garantia de armazenamento mais permanente aos dados e programas do usuário. Alguns dispositivos são diretamente ligados: disco rígido, outros são conectados quando necessário: disquetes, fitas de armazenamento, CD-ROM. São os mais lentos em comparação com os outros níveis de memória, mas possuem a maior capacidade de armazenamento e armazenam os dados de forma permanente.

PROCESSADOR XEON (fonte: www.clubedohardware.com.br)

O termo "Xeon" (pronuncia-se "zíon") foi incluído pela Intel aos seus processadores que são voltados para o mercado de servidores e estações de trabalho. Esses processadores reconhecem mais memória RAM, permitem trabalhar em ambiente multiprocessadro (isto é, com placas-mãe com vários processadores instalados sobre ela) e possui um desempenho maior que os processadores voltados para o mercado doméstico.

Uma visão geral sobre os processadores XEON:

Pentinum II Xeon

A principal diferença entre o Pentium II Xeon e o Pentium II é o clock em que o cache de memória L2 é acessado. Enquanto que o Pentium II Xeon acessa o seu cache L2 na mesma freqüência de operação interna (ex: 400 MHz em um Pentium II Xeon de 400 MHz), o Pentium II acessa o seu cache L2 na metade de sua freqüência de operação interna (ex: 200 MHz em um Pentium II de 400 MHz).

As principais características do Pentium II Xeon são as seguintes:

- *32 KB de cache L1 dividido, sendo 16 KB para instruções e 16 KB para dados.*
- *512 KB, 1 MB e 2 MB de cache de memória L2 sendo acessado na mesma freqüência de operação interna do processador.*
- *Barramento externo de 100 MHz.*
- *Acesso a até 64 GB de memória RAM.*

Pentium III Xeon

As principais características do Pentium III Xeon são as seguintes:

- *32 KB de cache L1 dividido, sendo 16 KB para instruções e 16 KB para dados.*
- *512 KB, 1 MB e 2 MB de cache de memória L2 sendo acessado na mesma freqüência de operação interna do processador.*
- *Barramento externo de 100 MHz ou 133 MHz.*
- *Acesso a até 64 GB de memória RAM.*
- *Multiprocessamento simétrico com até quatro processadores (os modelos com 2 MB de cache L2 permitiam multiprocessamento com até oito processadores).*

Pentium Xeon

Este processador deveria se chamar Pentium 4 Xeon, mas a Intel optou pelo nome Xeon. Como comentamos anteriormente, o Xeon é um processador voltado para o mercado de servidores e estações de trabalho baseado no Pentium 4, sendo, portanto, um processador Intel de 7ª geração. Como vimos, os processadores anteriores da série Xeon usavam a arquitetura Intel de 6ª geração (a mesma do Pentium Pro).

Os processadores Xeon possuem 8 KB de memória cache L1 para dados (16 KB nos modelos que possuem a tecnologia de 64 bits EM64T) e um cache L1 de execução de 150 KB. O cache L2 pode ser de 512 KB, 1 MB ou 2 MB, sendo que alguns modelos possuem um cache L3, que pode ser de 1 MB, 2 MB, 4 MB ou 8 MB.

XEON MP

Como já explicamos, a diferença entre o Xeon MP e o Xeon é a quantidade de processadores suportados no modo de multiprocessamento simétrico: o Xeon suporta até dois processadores em uma mesma placa-mãe, enquanto que o Xeon MP suporta até quatro processadores por barramento.

As principais características dos processadores Xeon MP são:

- *Soquete 603.*
- *Cache L1 de execução de 150 KB.*
- *Cache L1 de dados de 8 KB ou de 16 KB nos modelos com suporte à tecnologia EM64T (tecnologia de 64 bits).*
- *Multiprocessamento simétrico diretamente com até quatro processadores.*
- *Tecnologia Hyper-Threading.*

ARQUITETURAS MULTICORE XEON COMERCIAIS:

=> Xeon de 2 núcleos

A tecnologia de dois núcleos traz dois processadores inteiros dentro de um mesmo invólucro. Como os processadores Xeon de núcleo duplo modelos 50xx, 70xx e 71xx têm a tecnologia HyperThreading – que simula a existência de dois processadores em cada núcleo – o sistema operacional reconhece cada processador Xeon de núcleo duplo como sendo quatro processadores. Assim, em um servidor com dois processadores Xeon de núcleo duplo, o sistema operacional reconhecerá oito processadores (quatro núcleos, dois por pocessador, e dois processadores lógicos por núcleo).

Os processadores Xeon de dois núcleos das séries 50xx, 70xx e 71xx são baseados na microarquitetura do Pentium 4 (NetBurst) e por isso possuem a tecnologia HyperThreading, que não está presente na microarquitetura Core.

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.

Principais características dos Xeon das séries 50xx, 70xx e 71xx:

- *Tecnologia de dois núcleos*

- *Mesma arquitetura interna no Pentium 4 (NetBurst)*
- *Cache L1 de dados de 16 KB e cache de execução de 150 KB.*
- *Cache L2 de 2 ou 4 MB compartilhado*
- *Cache L3 interna de 4 MB, 8 MB ou 16 MB (somente nos modelos 71xx).*
- *Suporte a multiprocessamento simétrico com até dois processadores por placa-mãe.*

Principais características dos Xeon 30xx, 51xx e 72xx Microarquitetura Core

- *Tecnologia de dois núcleos*
- *Microarquitetura Core*
- *Tecnologia de fabricação de 65 nm*
- *Cache L1 dividido, sendo 32 KB para dados e 32 KB para instruções por núcleo*
- *Cache L2 de 4 MB compartilhado entre os núcleos*

Principais características dos Xeon 31xx e 52xx

- *Tecnologia de dois núcleos*
- *Microarquitetura Core*
- *Tecnologia de fabricação de 45 nm*
- *Cache L1 dividido, sendo 32 KB para dados e 32 KB para instruções por núcleo*
- *Cache L2 de 6 MB compartilhado*

=> Xeon de 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 na página anterior. Com isso, o cache L2 desses processadores não é compartilhado entre todos os seus núcleos: os núcleos 1 e 2 compartilham um mesmo cache L2, enquanto que os núcleos 3 e 4 compartilham um outro cache L2. O valor divulgado é o valor total (soma dos dois caches). Leia o nosso artigo Visão Geral dos Futuros Processadores de Quatro Núcleos da Intel para uma explicação mais detalhada sobre a arquitetura usada por estes processadores.

Principais características dos processadores Xeon das séries 32xx, 53xx e 73xx:

- *Microarquitetura Core*
- *Tecnologia de quatro núcleos*
- *Tecnologia de fabricação de 65 nm*
- *Cache L1 dividido, sendo 32 KB para dados e 32 KB para instruções por núcleo*
- *Cache L2 de 4 MB, 6 MB ou 8 MB, dependendo do modelo, dividido em dois*

Principais características dos processadores Xeon 33xx e 54xx:

- *Microarquitetura Core*
- *Tecnologia de quatro núcleos*
- *Tecnologia de fabricação de 45 nm*
- *Cache L1 dividido, sendo 32 KB para dados e 32 KB para instruções por núcleo.*
- *Cache L2 de 6 MB ou 12 MB dividido em dois*

5. (1,0) Considere uma máquina com 1 Giga células de memória onde cada célula armazena uma palavra e cada instrução tem o tamanho de uma palavra. Esta máquina possui um conjunto de instruções com 512 instruções distintas, sendo cada uma delas composta de um código de operação e um único operando, que indica o endereço de memória.

a) Qual o tamanho mínimo do REM?

REM = E = tamanho em bits necessários para acessar toda a memória (N)
N = 2^E = 1 G células = 2^{30} células => E = 30 => REM = 30 bits

b) Qual o tamanho mínimo do RI?

Tamanho mínimo de RI = tamanho da instrução
Tamanho da instrução = código de operação + 1 operando
Tamanho da instrução = (mínimo para endereçar 512 instruções) + (endereço de memória)
Tamanho da instrução = 9 bits + 30 bits = 39bits (tamanho mínimo)

c)Qual o tamanho mínimo do RDM?

Tamanho do barramento = tamanho da célula = tamanho da instrução
RDM = tamanho do barramento de dados = 39bits (tamanho mínimo)

d) Qual o tamanho da memória em bits?

T = M x N => T = tamanho da célula x memória principal =>
T = 39 bits/células x 1Giga células => T = 39 Giga bits

e) Se a largura do barramento de dados desta máquina for igual à metade do tamanho de uma instrução, como funcionará o ciclo de busca?

Seriam necessários 2 ciclos de busca para transferir uma instrução completa.

6. (0,5) 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) STR 100

a) RI <- Instrução lida
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 gravação em memória
d) Busca do operando na memória
- A UC emite sinais para que o valor do campo operando = 100 seja transferido para a REM
- A UC emite sinais para que o valor do registrador acumulador seja transferido para a RDM
- A UC ativa a linha WRITE do barramento de controle
- Conteúdo do RDM é transferido, através do barramento de dados, para o endereço 100 da memória, endereço este transferido do REM a partir do barramento de endereços.

b) JN 200

a) RI <- Instrução lida
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 salto condicional
d) UC emite sinal para transferir conteúdo acumulador para UAL
- UAL <- ACC
e) Executa operação de comparação
e.1) Resultado = verdadeiro, isto é, ACC < 0
CI <- Operando (CI <- 200)
d) Inicia o procedimento de leitura da instrução contida no endereço que consta em CI

7. (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 20 é lido e verifica-se se o seu valor é 0. Caso seu valor seja 0, o conteúdo de memória cujo endereço é 30 é somado ao conteúdo de memória cujo endereço é 40 e o resultado é armazenado no endereço 50. Caso contrário, o conteúdo de memória cujo

endereço é 30 é subtraído do conteúdo de memória cujo endereço é 40 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 4bits.

Endereço	***Instrução***	***Descrição***	***Linguagem Máquina (bin / hexa)***	
Hex ou dec			*Considerando Endereços em hexa*	*Considerando Endereços em decimal*
10	*LDA 20*	*ACC <- (20)*	*(000100100000 / 120)*	*(000100010100 / 114)*
11	*JZ 16*	*CI <- 16 se ACC=0*	*(010100010110 / 516)*	*(010100010000 / 510)*
12	*LDA 40*	*ACC <- (40)*	*(000101000000 / 140)*	*(000100101000 / 128)*
13	*SUB 30*	*ACC <- ACC - (30)*	*(010000110000 / 430)*	*(010000011110 / 41E)*
14	*STR 50*	*(50) <- ACC*	*(001001010000 / 250)*	*(001000110010 / 232)*
15	*HLT*	*Encerra*	*(000000000000 / 000)*	*(000000000000 / 000)*
16	*LDA 30*	*ACC <- (30)*	*(000100110000 / 130)*	*(000100011110 / 11E)*
17	*ADD 40*	*ACC <- ACC + (40)*	*(001101000000 / 340)*	*(001100101000 / 328)*
18	*STR 50*	*(50) <- ACC*	*(001001010000 / 250)*	*(001000110010 / 232)*
19	*HLT*	*Encerra*	*(000000000000 / 000)*	*(000000000000 / 000)*

Outra solução,

Endereço	***Instrução***	***Descrição***	***Linguagem Máquina (bin / hexa)***	
Hex ou dec			*Considerando Endereços em hexa*	*Considerando Endereços em decimal*
10	*LDA 20*	*ACC <- (20)*	*(000100100000 / 120)*	*(000100010100 / 114)*
11	*JZ 15*	*CI <- 15 se ACC=0*	*(010100010111 / 517)*	*(010100010001 / 511)*
12	*LDA 40*	*ACC <- (40)*	*(000101000000 / 140)*	*(000100101000 / 128)*
13	*SUB 30*	*ACC <- ACC - (30)*	*(010000110000 / 430)*	*(010000011110 / 41E)*
14	*JMP 17*	*CI <- 17*	*(100000010111 / 817)*	*(100000010001 / 811)*
15	*LDA 30*	*ACC <- (30)*	*(000100110000 / 130)*	*(000100011110 / 11E)*
16	*ADD 40*	*ACC <- ACC + (40)*	*(001101000000 / 340)*	*(001100101000 / 328)*
17	*STR 50*	*(50) <- ACC*	*(001001010000 / 250)*	*(001000110010 / 232)*
18	*HLT*	*Encerra*	*(000000000000 / 000)*	*(000000000000 / 000)*

OBS: no quadro acima, foram abordados os entendimentos relativos aos endereços, que podiam ser considerados tanto como em decimal ou hexadecimal

8_ (1,2) Considere uma máquina com arquitetura semelhante àquela apresentada em aula. Pode-se endereçar no máximo 256 M células de memória, sendo que cada célula tem tamanho igual a 8 bits. Em cada acesso á memória, obtém-se o conteúdo de uma célula.Todas as instruções desta máquina possuem três campos: o primeiro indica o código de operação e o segundo e terceiro indicam endereços de célula de memória onde se encontram os operandos. Esta máquina possui 200 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 = 256M células = 2^{28} células => E = 28 => REM = 28 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 = 28 bits

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

Tamanho mínimo de RI = tamanho da instrução

Tamanho da instrução = código de operação + 2 operandos
Tamanho da instrução = (tamanho p/ endereçar 200 instruções) + 2 x (endereço de memória)
Tamanho da instrução = 8 bits + 2 x 28 bits = 64 bits
Tamanho mínimo de RI = 64bits

d) Calcule o número de células que uma instrução ocupa.
Tamanho mínimo de 1 instrução = 64 bits
Tamanho de 1 célula = 8 bits
Serão necessárias pelo menos 8 células para uma instrução

e) Calcule a capacidade máxima de armazenamento da memória deste sistema em bits.
T = N x M => T = 256 Mcélulas x 8bits/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 acessar toda a memória = 28 bits

9. (0,8) Considere uma máquina cujo relógio possui uma freqüência de 1 GHZ e um programa no qual são executadas 100 instruções desta máquina.

a) Calcule o tempo para executar este programa, considerando que cada instrução é executada em 10 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 = 1/1.000.000.000 = 0,000 000 001 seg ou 1ns (nanosegundos)
Tempo de execução de 1 instrução = 10 ciclo de relógio = 10 x 1ns = 10ns
*100 instruções executadas seqüencialmente = 100 x 10ns = **1000ns***

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 este programa, considerando que não existem conflitos de qualquer tipo.

Tempo para um estágio = 2 ciclos de relógio = 2 x 1ns = 2ns
Para execução da 1ª instrução = 5 estágios x 2ns = 10ns
Para execução das instruções posteriores = tempo de 1 estágio devido ao pipeline = 2ns
*Tempo total para execução das 100 instruções = 10ns + 99 x 2ns = **208ns***

c) Uma segunda nova implementação dessa máquina utiliza um pipeline de 6 estágios, todos de duração igual a 1 ciclo de relógio. Calcule o tempo para executar este programa, considerando que não existem conflitos de qualquer tipo.

Tempo para um estágio = 1 ciclos de relógio = 1 x 1ns = 1ns
Para execução da 1ª instrução = 6 estágios x 1ns = 6ns
Para execução das instruções posteriores = tempo de 1 estágio devido ao pipeline = 1ns
*Tempo total para execução das 100 instruções = 6ns + 99 x 1ns = **105ns***

10. (0,5) Calcule o tempo MÍNIMO para transmitir um arquivo de 8 Megabytes armazenado em um servidor WEB para a sua máquina, considerando os seguintes modos de transmissão:

Para considerar o tempo mínimo, devemos considerar a maior taxa de transmissão possível, sem considerar bits adicionais (a não ser pelo item b), retransmissão, etc..

a) Modo síncrono com taxa de 56 Kbits por segundo.

Taxa de transmissão = 56 Kbits/segundo = 56.000 bits / segundo
Arquivo a ser transmitido = 8 MBytes = 8 x 1024 x 1024 bytes x 8 bits/byte = 67.108.864 bits
Tempo necessário = 67.108.864 / 56.000 = 1198,3 segundos

b) Modo assíncrono com taxa de 19.2 Kbits por segundo. Considere que neste caso para cada byte a ser enviado necessita-se do envio de um bit para indicar inicio de byte e outro bit para indicar fim de byte.

Taxa de transmissão = 19.2 Kbits/segundo = 19.200 bits / segundo
Arquivo a ser transmitido = 8 MBytes = 8 x 1024 x 1024 bytes x
(8 bits/byte + 1 bits_inicio/byte + 1bit_fim/byte) = 83.886.080 bits
Tempo necessário = 83.886.080 / 19.200 = 4369 segundos

11) (1,0) Explique detalhadamente como funciona uma unidade de controle microprogramada e explique as diferenças existentes entre microinstruções verticais e 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 simples e especificam microoperações. Uma unidade de controle microprogramada é implementada com circuitos lógicos e é capaz de executar uma seqüência de microinstruções que geram sinais de controle para o funcionamento da UCP. Estes sinais de controle são utilizados para causar transferências de dados entre registradores e memória e execução de operações pela ULA, por exemplo.

As microinstruções horizontais tem como característica gerar sinais de controle distintos para a UCP diretamente de cada bit que as compõem. A implementação de microinstruções horizontais apresenta a vantagem de ser simples e direta, podendo controlar várias microoperações em paralelo, além de uma eficiente utilização do hardware. Possui a desvantagem de maior ocupação de espaço de memória de controle onde as microinstruções são armazenadas em relação à microinstrução vertical.

As microintruções verticais se caracterizam por possuírem menos bits que os sinais necessários para o controle da UCP. Estes sinais de controle são gerados através da decodificação dos bits das microinstruções verticais. Sua principal vantagem é reduzir a área de armazenamento na memória de controle em função do menor tamanho da microinstrução, mas poderá ser necessária uma maior quantidade de instruções. Tem como principal desvantagem o aumento no tempo de geração dos sinais de controle devido à necessidade da decodificação dos campos de cada microinstrução.