

# Importantes para a P1

Wednesday, April 25, 2018 8:23 AM

## CAPÍTULO 1: (IMPORTANTE)

**Arquitetura:** refere-se aos atributos visíveis para o programador (e.g., existe uma instrução de multiplicação?).

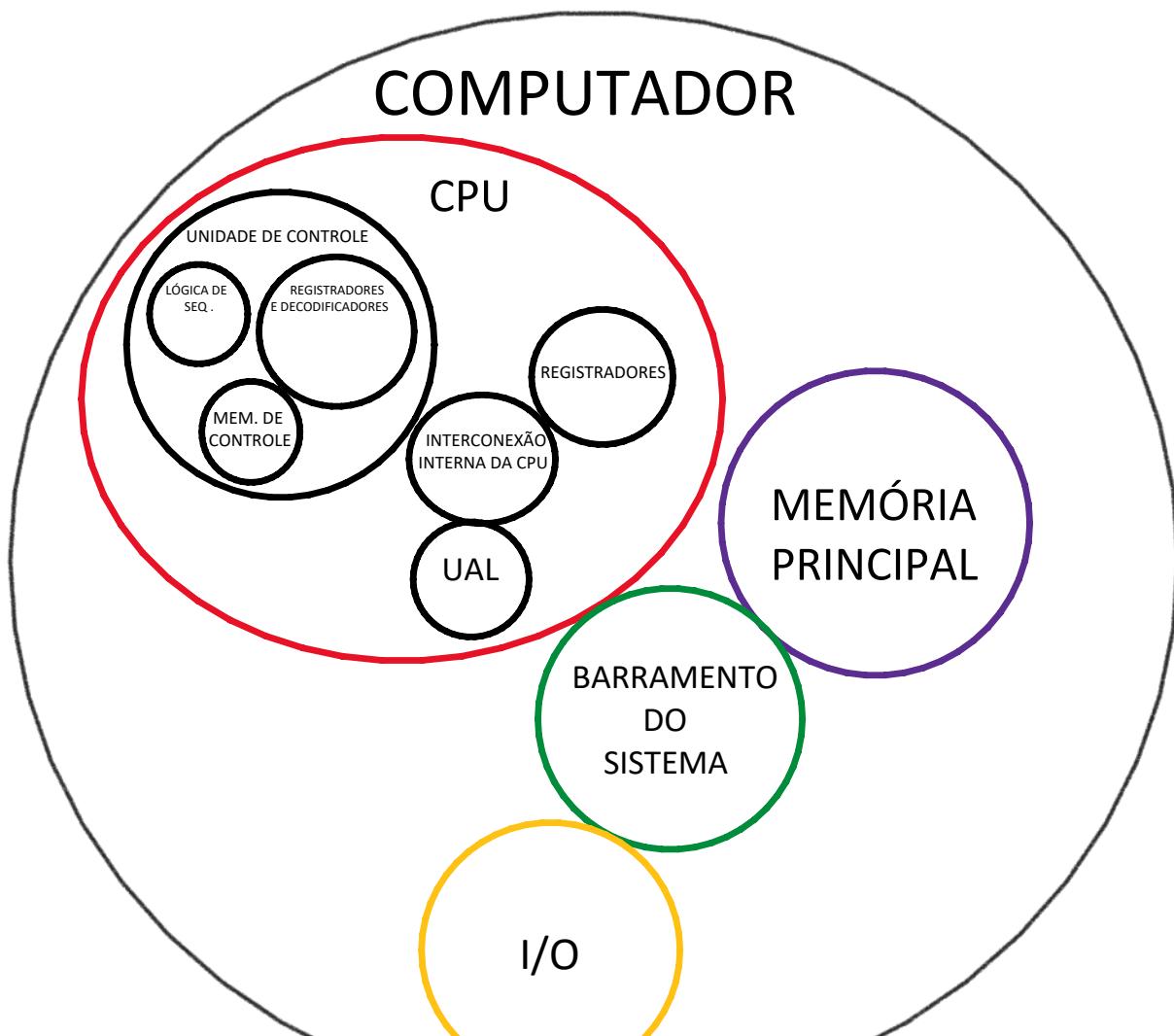
**Organização:** refere-se à maneira como esses atributos são implementados (e.g., a instrução de multiplicação é implementada por hardware ou é feita por somas repetidas?).

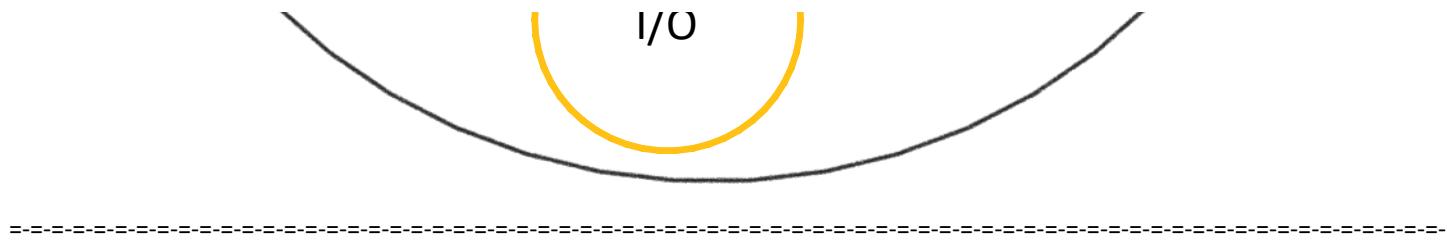
**Computador:** é um sistema hierárquico, i.e. constituído de um conjunto de subsistemas interrelacionados, cada um contendo outros subsistemas.

**Estrutura:** refere-se à maneira como os componentes relacionam-se uns com os outros.

**Função:** refere-se à operação dos componentes individualmente enquanto parte da estrutura.

- Processamento de dados
- Armazenamento de dados
  - o Temporário
  - o Permanente
- Transferência de dados
  - o Interna
  - o Externa (periféricos e comunicação de dados)
- Controle (e coordenação das funções acima citadas)





## CAPÍTULO 2:

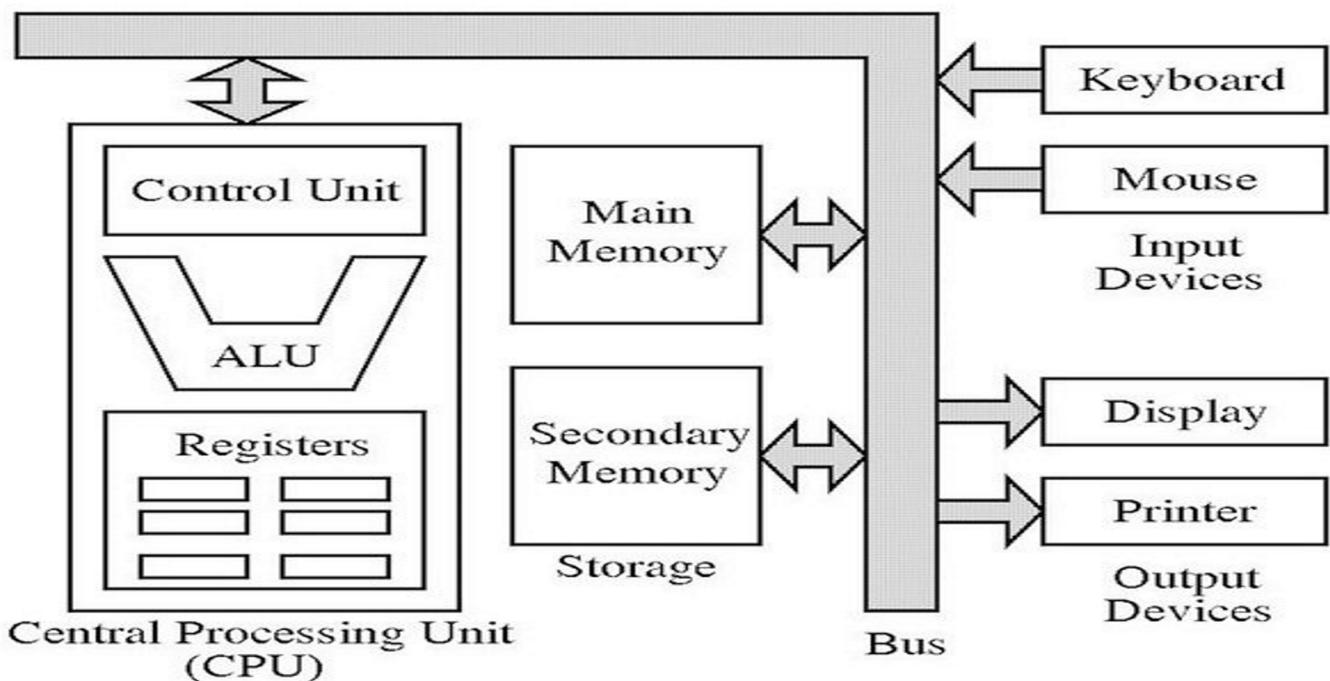
**Primeira Geração (1951-1959):** funcionavam por meio de circuitos e válvulas (Ex.: ENIAC)

**Segunda Geração (1959-1965):** funcionavam por meio de transistores (Ex.: IBM 7000)

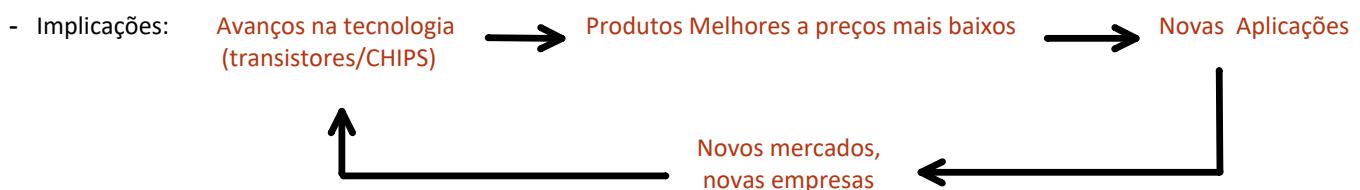
**Terceira Geração (1965-1975):** funcionavam por circuitos integrados (Ex.: IBM 360)

**Quarta Geração (1975-até os dias atuais):** computadores diminuem de tamanho, aumentam a velocidade e capacidade de processamento de dados. São incluídos os microprocessadores com gasto cada vez menor de energia.

**Estrutura da máquina de von Neumann:**



**Lei de Moore:** o poder de processamento dos computadores (entenda computadores como a informática geral, não os computadores domésticos) dobraria a cada 18 meses.



## CAPÍTULO 3:

**Ciclo de Instrução (IMPORTANTE)**

- Dois passos:

o Busca

- Ciclo de Busca

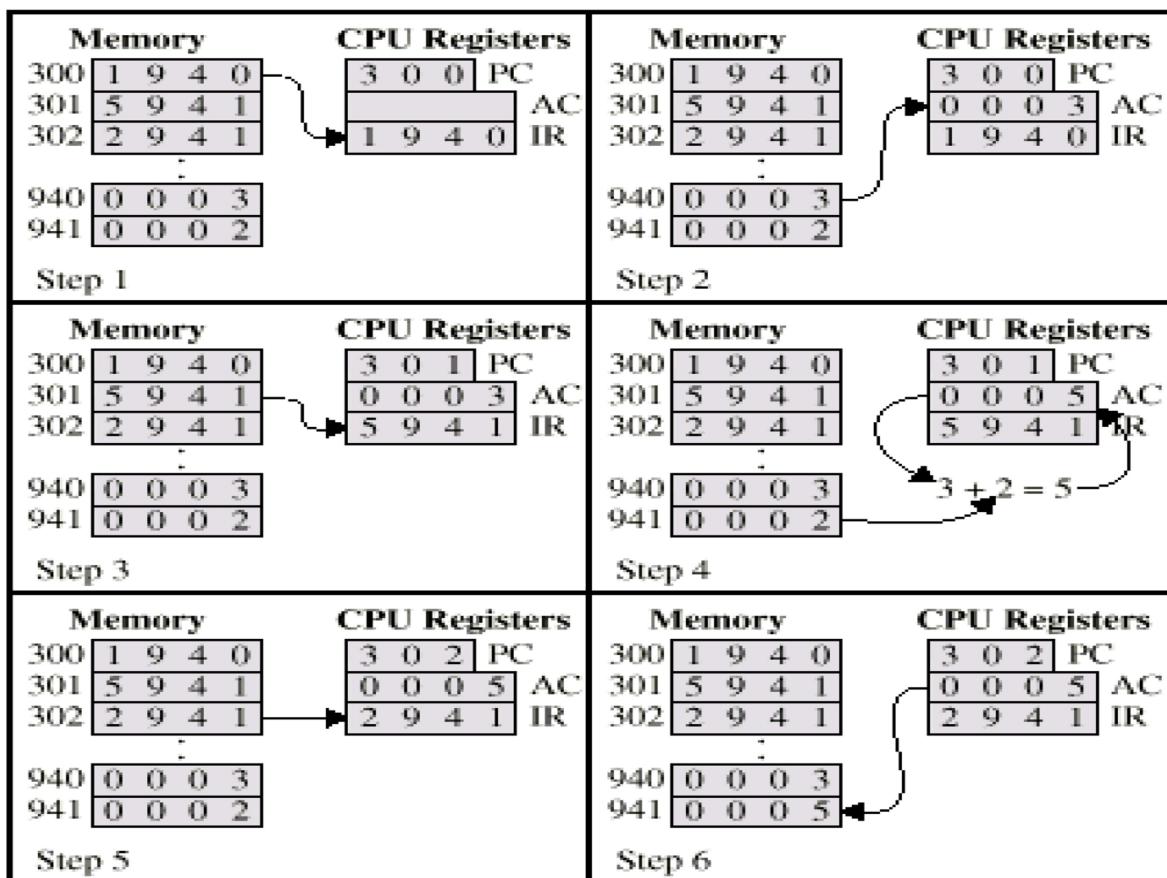
- programa Counter (PC) guarda o endereço da próxima instrução a ser buscada
- Processador busca a instrução da localização de memória apontada pelo PC
- Incremento do PC
  - o A não ser que se diga o contrário
- Instrução é carregada no Instruction Register
- Processador interpreta a instrução e realiza as ações necessárias

o Execução

- Ciclo de Execução

- Processador-memória
  - o Transferência de dados entre CPU e memória principal
- Processador - I/O
  - o Transferência de dados entre CPU e o módulo de I/O
- Processamento de Dados
  - o Alguma operação aritmética ou lógica nos dados
- Controle
  - o Alteração da sequência de operações
  - o ex. jump
- Combinação das etapas anteriores

(ex de execução de um programa):



### Interrupções: (IMPORTANTE)

- Mecanismo pelo qual outros módulos (ex. I/O) podem interromper sequência normal de processamento o programa
  - o ex. overflow, divisão por zero
- Temporizador (timer)
  - o Gerado pelo temporizador interno do processador

- o Usada na multi-tarefa preemptiva
- I/O
  - o Vinda da controladora de I/O
- Falha de Hardware
  - o ex. Erro de paridade de memória
- Ciclo de Interrupção
  - o Adicionado ao ciclo de instrução
  - o Processador verifica se houve interrupção
    - Indicada por um sinal de interrupção
  - o Se não há interrupção, busca a próxima instrução
  - o Se há interrupção pendente:
    - Suspende a execução atual do programa
    - Salva o contexto
    - Coloca o PC no endereço inicial da rotina de manipulação da interrupção
    - Processa a interrupção
    - Restaura o contexto e continua o programa interrompido
- Interrupções Múltiplas
  - o Desabilita interrupções
    - processador ignorará outras interrupções enquanto isso processamento de uma interrupção
    - interrupções permanecem pendentes e serão verificadas depois que a primeira interrupção seja processada
    - interrupções são tratadas na sequência que ocorrem
  - o Define prioridades
    - Interrupções de baixa prioridade podem ser interrompidas por interrupções de mais alta prioridade
    - Quando a interrupção de alta prioridade foi processada,
    - processador retorna à interrupção anterior

### O que é um barramento?

- Um meio de comunicação conectando dois ou mais dispositivos
- Geralmente fazem broadcast
- Geralmente agrupados
  - o Vários canais em um barramento
  - o ex. Barramento de dados de 32 bit são 32 canais de um bit separados

### Barramento de Dados

- Carrega dados
  - o Ainda não há diferença entre “dados” e “instrução” neste nível
- Largura é chave determinante do desempenho
  - o 8, 16, 32, 64 bit

### Barramento de Endereço

- Identifica a fonte ou destino dos dados
  - o ex. CPU necessita ler uma instrução (dado) de uma dada localização na memória
- Largura do barramento determina a capacidade máxima do sistema

### Barramento de Controle

- Informação de controle e temporização
  - o Sinal de leitura/escrita de memória
  - o Requisição de interrupção
  - o Sinais de Clock

### Tipos de barramento

- Dedicado
  - o Separa linhas de dados & endereços
- Multiplexados
  - o Linhas compartilhadas
  - o Linha de endereço válido ou linha de controle de dados válidos

- o Vantagem – menos linhas
- o Desvantagens
  - Controle mais complexo
  - Desempenho

### **Problemas de Barramentos Únicos**

- Vários dispositivos em um único barramento leva a:
  - o Atraso de propagação
    - Grandes caminhos significam que a coordenação do barramento pode afetar desfavoravelmente o desempenho.

### **Arbitração de barramento**

- Mais de um módulo controlando o barramento
- Apenas um módulo pode controlar o barramento por vez
- Arbitração pode ser centralizada (Um único dispositivo de hardware controlando acesso ao barramento) ou distribuída (Cada módulo pode pedir o barramento).

**Temporização (Timing):** coordenação de eventos no barramento.

---

## **CAPÍTULO 4: (IMPORTANTE)**

### **MEMÓRIA:**

#### **Localização:**

- A memória interna refere-se à memória principal, diretamente acessada pela CPU.
- A CPU necessita da sua própria memória local, na forma de registradores.
- A memória externa (memória secundária) é referida a todos os dispositivos usados para armazenar dados e que acessem a CPU através de controladores/módulos de Entrada/Saída.

#### **Métodos de Acesso:**

- Sequencial
  - o Começa no início e lê os dados sequencialmente
  - o O tempo de acesso depende da localização dos dados e da posição anterior
  - o Por exemplo, uma fita cassete
- Direto
  - o Blocos individuais têm endereços únicos
  - o O acesso é feito por salto e por procura sequencial
  - o O tempo de acesso depende na localização dos dados e da localização anterior
  - o Por exemplo, um disco
- Aleatório
  - o Endereços individuais identificam locais exatos
  - o O tempo de acesso é independente do local ou de acessos anteriores
  - o Por exemplo, a RAM
- Associativa
  - o Os dados são localizados por comparação de parte dos dados armazenados
  - o O tempo de acesso é independente do local ou de acesso anteriores
  - o Por exemplo, a cache

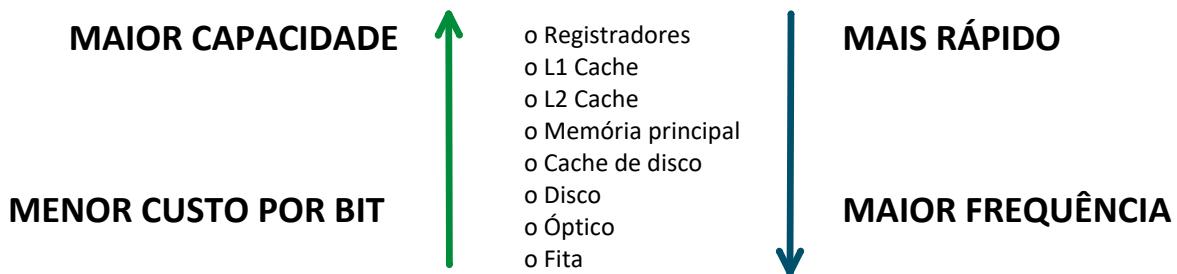
#### **Desempenho:**

- Tempo de ciclo de memória (Memory cycle time)
  - o Compreende o tempo de acesso e o tempo necessário a memória para recuperar antes que outro acesso possa ser realizado
  - o  $\text{Tempo de ciclo de memória} = \text{acesso} + \text{recuperar}$
- Taxa de transferência (Transfer Rate)
  - o Taxa a qual os dados podem ser transferidos
  - o Para memória aleatórias é  $1/(\text{Tempo de Ciclo})$

o Para as outras memórias:  $TN = TA + N/R$

- TN - Tempo médio de leitura ou escrita de N bits
- TA - Tempo médio de acesso
- N - Número de bits
- R - Taxa de transferência, em bits por segundo (bps)

Hierarquia de memória:

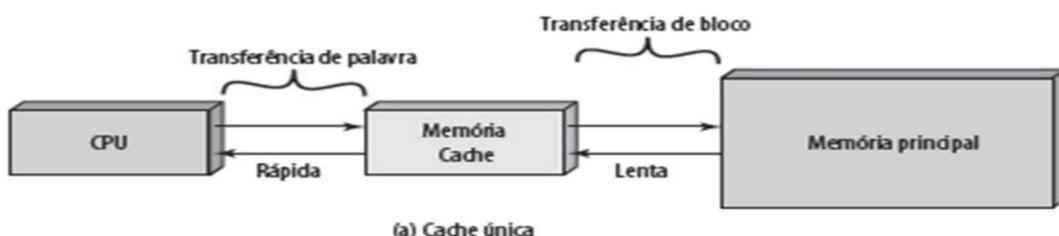


#### MEMÓRIA CACHE:

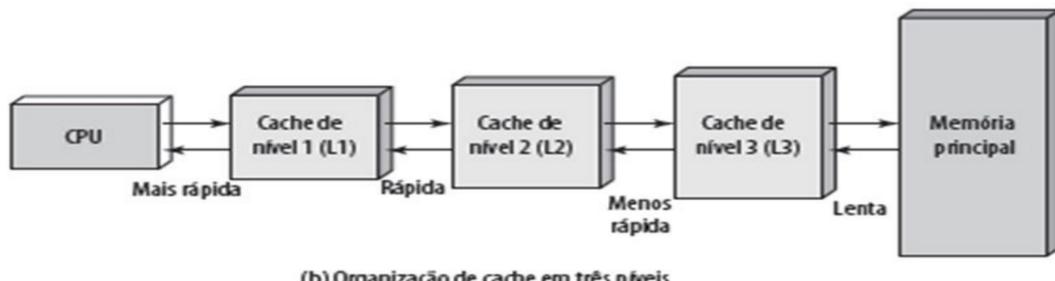
- O tempo de execução de uma instrução é inferior a tempo de acesso à memória
- A taxa que o CPU pode executar as instruções é claramente limitada pelo ciclo de acesso à memória
- Como o tempo de acesso à memória não acompanha o tempo de ciclo do processador, a solução consiste em colocar uma memória pequena e rápida entre o processador e a memória principal
- Esta memória é denominada memória cache.

Cache:

- Pequena memória rápida
- Posiciona-se entre a memória principal e a CPU
- Pode estar localizada na CPU ou num módulo



(a) Cache único



(b) Organização de cache em três níveis

#### Operações da Cache:

- o A CPU pede dados a memória principal
- o A cache é pesquisada a procura dos dados
- o Se estes estão presentes, os dados são obtidos rapidamente
- o Se não estão presentes, um bloco de dados é lido da memória principal para a cache
- o Os dados são então entregues da cache a CPU

- ↓
- o Se estes estão presentes, os dados são obtidos rapidamente
  - o Se não estão presentes, um bloco de dados é lido da memória principal para a cache
  - o Os dados são então entregas da cache a CPU
  - o A cache inclui identificadores (tags) para identificar os blocos de memória principal presentes na cache

#### Estrutura da Cache:

- Para uma memória principal de  $2n$  palavras endereçadas, dividida em blocos de  $K$  palavras ( $M=2n/K$  blocos), a memória cache é organizada em  $C$  slots de tamanho fixo que contêm  $K$  palavras cada
- Como é óbvio  $C << M$
- Por razões de eficiência, as leituras para (e da) cache são feitas em bloco e não palavra a palavra
- Como existem mais blocos de memória que slots ( $C << M$ ), um slot individual não pode estar permanentemente dedicado a um bloco
- Então cada slot inclui uma tag que identifica o bloco que está ser guardado
- Esta tag é usualmente parte do endereço da memória principal

#### Mapeamento Direto:



- Tamanho da cache = 32KB
- Tamanho do bloco = 16 bytes
  - o i.e. cache tem  $2K$  (211) linhas de 16 bytes
- Memória principal = 16 MB
  - o Endereços de 24 bit ( $2^{24}=16M$ )
  - o  $2^{20}$  (1M) blocos de 24 bytes
- Cada bloco da memória principal é mapeado a uma só linha da cache
  - o i.e. se um bloco está na cache, este tem de estar numa posição específica
- Fórmula para mapear um bloco de memória a uma linha de cache :  $i = j \bmod c$ 
  - o  $i$ = Número da Linha da Cache
  - o  $j$ = Número do Bloco da Memória Principal
  - o  $c$ = Número de Linhas na Cache
- Os  $w$  bits menos significativos identificam uma palavra numa linha da cache
- Os próximos  $s$  bits significativos identificam em que slot um endereço é mapeado
- Os últimos  $t$  bits significativos identificam um bloco de memória

(EX.: com  $C = 4$ ):

o Linha da Cache		Bloco de Memória Principal
o 0		0, 4, 8, ...
o 1		1, 5, 9, ...
o 2		2, 6, 10, ...
o 3		3, 7, 11, ...

o Regra Geral:

o 0		0, C, 2C, 3C, ...
o 1		1, C+1, 2C+1, 3C+1, ...
o 2		2, C+2, 2C+2, 3C+2, ...
o 3		3, C+3, 2C+3, 3C+3, ...

**8**  
**Tag s - r**

**14**  
**Linha (slot) r**

**2**  
**Palavra w**

- Endereços de 24 bit
- Identificador de palavra: 2 bits (blocos de 4 bytes)
- Identificador de bloco: 22 bit
  - o Tag de 8 bit (=22-14)
  - o Linhas de 14 bit
- Dois blocos na mesma linha nunca têm o mesmo Tag
- Encontram-se dados na cache procurando a linha (r) e comparando a Tag (s-r)

#### Mapeamento Direto prós & contras:

- Simples
- Barato
- Localização fixa para os blocos
  - o Se um programa acede a dois blocos que mapeiam na mesma linha repetidamente, os cache misses tornam-se muito elevados

#### **Mapeamento Associativo:**

- Um bloco de memória pode ser carregado em qualquer linha da cache
- Um endereço de memória é composto por tag e palavra
- O tag identifica o bloco de memória
- Cada linha de uma tag é examinada a procura de um match
- A pesquisa da cache é cara
  - o Idealmente seria necessário um circuito que pode simultaneamente examinar todas as tags ao mesmo tempo
- Implicaria a utilização de muitos circuitos
  - o custo elevado
- Como qualquer dado pode ser expulso da cache é necessário desenvolver regras de substituição

#### **Mapeamento Associativo por Conjuntos:**

- Divide a cache em v conjuntos de k linhas
  - o  $m = v \times k$
  - o  $i = j \text{ modulo } v$ , onde
    - i = número do conjunto na cache
    - j = número do bloco na memória principal
    - m = número de linhas da Cache
- Com este tipo de mapeamento, o bloco B pode ser mapeado em qualquer linha do conjunto i
- Permite capturar as vantagens dos dois métodos anteriores
  - o Nos casos extremos se  $v = m$ ,  $k = 1$ (direta); se  $v = 1$ ,  $k = m$ (associativa)

#### **Algoritmos de Substituição:**

- Qual o bloco a substituir? (se o bloco não está na cache)
- Mapeamento Direto
  - o Só existe uma opção
- Mapeamento Associativo
  - o Cada bloco é um candidato
- Associativa de nível k
  - o Um de k
- A substituição de blocos na cache pode ser implementada por diversos algoritmos
  - o LRU (least recently used)
  - o FIFO (First-in First Out)
  - o LFU (least frequently used)
  - o Random (aleatoriamente)

#### **Política de Escrita:**

- Existem duas técnicas de escrita de blocos modificados na memória principal
  - o Write-Through - as operações são executadas em simultâneo na cache e na memória
  - o Write-back - as operações são executadas apenas na cache e atualizando a memória principal apenas quando o bloco é substituído

#### Caches unificadas versus separadas:

- Uma cache para dados e instruções ou duas, uma para dados e uma para instruções.
  - Vantagens da cache unificada:
    - o Maior taxa de acerto.
      - Equilibra carga entre buscas de instrução e dados.
      - Apenas uma cache para projetar e implementar.
  - Vantagens da cache separada:
    - o Elimina disputa pela cache entre a unidade de busca/decodificação de instrução e a unidade de execução.
      - Importante no pipeline de instruções.
- 

## CAPÍTULO 5: (IMPORTANTE)

#### Memória Semicondutora:

##### Random Access Memory (RAM):

- Leitura/Escrita
- Volátil
- Armazenamento de dados temporário
- A tecnologia RAM encontra-se dividida em duas categorias:
  - o RAM Dinâmica
    - As células guardam os dados em capacitores, a presença ou ausência de carga no capacitor indica o nível lógico (1 ou 0)
    - Como os capacitores perdem a sua carga, este tipo de memória necessita de ser refrescada periodicamente
    - São de construção simples
    - São pequenos
    - Custam menos
    - Mais lentas
    - Utilização -> Memória principal
  - o RAM Estática
    - Neste tipo de RAM os bits são guardados em configurações lógicas do tipo flip-flop
    - Não necessitam de refrescamento
    - Construção mais complexa
    - São mais volumosos
    - São mais caros
    - Mais rápidos
    - Utilização -> Cache
    - As RAM estáticas com bateria dão-se normalmente a designação de RAM não voláteis (NVRAM)

##### Read Only Memory (ROM):

- Às memórias em que só é possível efectuar operações de leitura é comum dar-se a designação de ROM (Read Only Memory)
- O seu conteúdo já está programado de fábrica
  - o Muito caras para pequenos volumes
- Mas o seu conteúdo pode ser programado pelo usuário
  - o PROM e EPROM
  - o Neste caso o termo mais correcto para este tipo de memória deveria ser PROM (Programmable ROM) ou EPROM (erasable PROM), respectivamente para referir os dispositivos que podem ser programados apenas uma vez, ou várias vezes
- Tipos de ROM:
  - o PROM (Programmable ROM)

- Podem ser programadas apenas uma vez
- Para a sua programação é necessário equipamento específico
- o EPROM (erasable PROM)
  - Podem ser programadas várias vezes
  - O conteúdo é apagado por UV
- o Electrically Erasable (EEPROM)
  - Demora consideravelmente mais tempo a escrever do que a ler
- o Flash memory
  - Apaga toda a memória elétricamente

#### **Detalhes de Organização:**

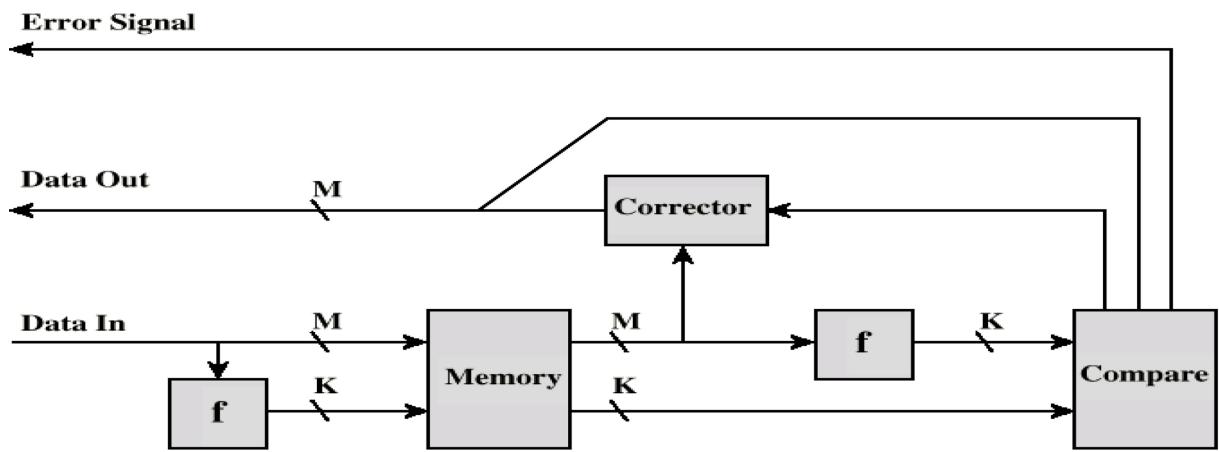
- elemento básico de uma memória semicondutora é a célula de memória que tem as seguintes propriedades:
  - o Exibem dois estados
  - o Podem ser escritas (pelo menos 1 vez) para alterar o seu estado
  - o Podem ser lidas para detectar o seu estado
- Chips de 16Mbit podem estar organizados em 1M entradas de palavras de 16 bit
- Chips de 16Mbit podem estar organizados numa matriz de 2048 x 2048 x 4bit
  - o Reduz o número de pinos para os endereços
  - o Multiplexar linhas de endereços e colunas de endereços
  - o 11 pinos de endereços ( $2^{11}=2048$ )
  - o Adicionar mais um pino quadruplica a capacidade de memória
- Para as memórias semicondutores, uma das chaves para o desenho é o número de bits de dados que podem ser lidos/escritos por acesso
- A memória está organizada em W palavras de b bits cada
- A memória DRAM está logicamente organizada em b matrizes de n xn elementos (onde a dimensão da memória é n xn xb)
- Por exemplo, para um chip de 16 Mbits DRAM em que 4 bits são lidos ou escritos de cada vez, 4 matrizes de 2.048 por 2.048 elementos existem
- Vários arranjos físicos são possíveis
- Os elementos estão ligados pelas linhas horizontais e verticais

#### **Refrescamento:**

- O circuito de refrescamento está incluído no chip
- Desativar o chip
- Percorrer as colunas
- Demora tempo
- Reduz o desempenho

#### **Deteção e correcção de erros:**

- A memória semicondutora está sujeita a erros que podem ser categorizados em:
  - o Falhas do tipo 'Hard' (grave)
    - Defeito físico permanente que impede a célula de armazenar dados. São provocados por ambientes hostis, defeitos de fabricação e desgaste
  - o Falhas do tipo 'Soft' (moderada)
    - Aleatórias, não destrutivas
    - Erros aleatórios, não destrutivos que alteram
    - conteúdo de uma ou mais células de memória sem danificarem a memória.
  - o Detecção usando código Hamming de correção de erros
- Quando uma palavra é lida, um código é usado para detectar e possivelmente corrigir erros
- Um novo conjunto K de bits de código são gerados a partir dos M bits da dados e comparados com os códigos de bits adquiridos
- Esta comparação retorna 1 dos 3 resultados:
  - o Não foram detectados erros. Os bits adquiridos são enviados;
  - o Foi detectado um erro e é possível corrigi-lo. Os bits de dados mais os bits de paridade foram introduzidos num corretor, que irá produzir uma nova sequência de M bits para serem enviados;
  - o Foram detectados erros, mas não é possível corrigi-los.



## CAPÍTULO 6:

### Disco magnético:

- Um disco é composto por um conjunto de pratos circulares em metal ou em plástico revestido com um material magnetizável (óxido de ferro)
- Os dados são escritos e lidos através de um dispositivo metálico (cabeça) que é fixo em relação ao prato.
- **RAID**
  - o Redundant Array of Independent Disks
  - o Redundant Array of Inexpensive Disks
  - o O sistema RAID consiste em 7 níveis (0-6)
    - RAID 0: Sem redundância / Os dados estão organizados por todos os discos / Aumenta a velocidade
    - RAID 1: Os dados são armazenados por vários discos
    - RAID 2: Os discos encontram-se sincronizados
    - RAID 3: Taxas de transferências muito altas
    - RAID 4: Faixas largas
    - RAID 5: Distribuição dos bits de paridade pelo vários discos / Muito usado em servidores de rede / Leitura e escrita independente
    - RAID 6: Similar ao 4 e o 5
  - o Um conjunto de discos físicos vistos como um único disco lógico pelo Sistema Operacional
  - o Os dados encontram-se distribuídos pelos discos físicos

## CAPÍTULO 7:

### O Barramento do Sistema:

- Para além da CPU e da memória, o terceiro elemento de um sistema de computador é o conjunto de módulos de Entradas e Saídas
- Cada módulo liga-se com ao barramento (bus) do sistema ou com o comutador central e controla um ou mais dispositivos periféricos
- O bus do sistema é usado para ligar dois ou mais elementos do sistema no computador
- Conjunto de linhas paralelas que transportam sinais usados para comunicação entre elementos

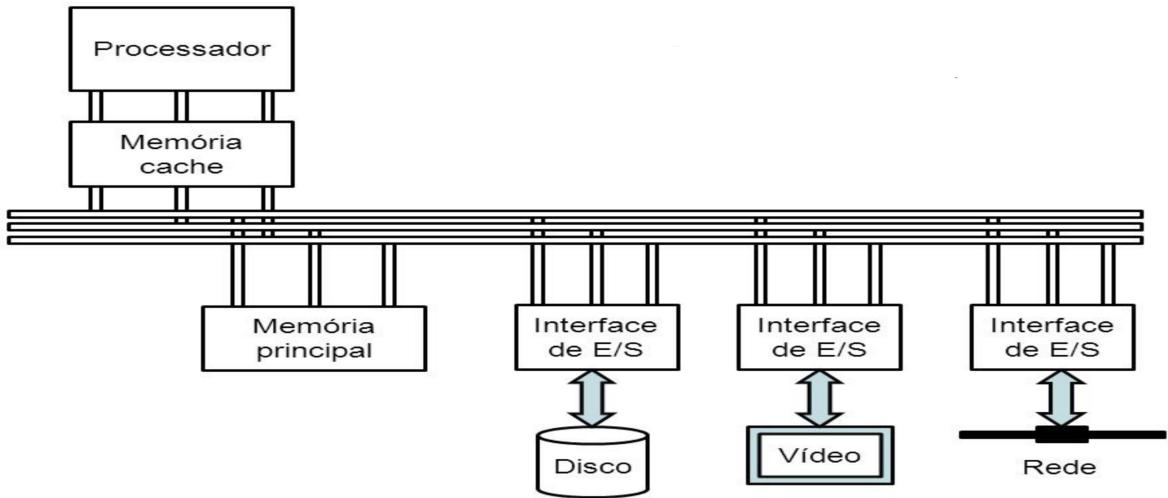


#### Módulos de I/O (E/S):

- Os módulos de I/O são dispositivos que possuem alguma “inteligência”
- Possuem funcionalidades lógicas capazes de estabelecer mecanismos de comunicação entre os diferentes periféricos e o bus de sistema
- Funções:
  - o Controle e Temporização
    - Controla o tráfego de dados entre os recursos internos e os dispositivos externos
  - o Comunicação com o CPU
    - Descodificação de comandos, transferência de dados, relato de estado e reconhecimento de endereços
  - o Comunicação com os dispositivos
    - Comandos, informação de estado e transferência de dados
  - o Buffering de dados
    - Espaços que permitem manter o ritmo de transferência
    - A CPU e a memória tem taxas de transferência de dados elevadas quando comparado com os dispositivos de periféricos
  - o Detecção de Erros
    - Relato de erros ao CPU

#### Problemas relacionados com os Dispositivos de I/O (E/S):

- Cada módulo possui uma interface com o bus do sistema e controla um ou mais dispositivos
- Os dispositivos de entradas e saídas não são simples conectores que se ligam diretamente ao bus do sistema
- As razões para que isto suceda são as seguintes:
  - o Diferentes níveis de sinal elétrico
  - o Velocidade
  - o Sincronização
  - o Códigos e formatos
- Diferentes níveis de sinal elétrico: Muitos equipamentos periférico não funciona com os níveis internos de um computador
- Velocidade: A maior parte dos periféricos tem escalas de tempo de ordens de grandeza abaixo dados processadores.
- Sincronização: Os periféricos não estão sincronizados com os processadores. Tem que existir protocolos de funcionamento que aliviem o processador da necessidade de se sincronizar com periféricos
- Códigos e formatos: Nem todos os periféricos usam os códigos que o processador utiliza para codificar os vários tipos de informação e, do mesmo modo, os formatos em que a informação é utilizada por cada um pode é diferente entre si e do processador
- Controle do funcionamento: Por fim é útil dispensar o processador das tarefas de controle detalhado dos periféricos (manipular as partes mecânicas dos periféricos, etc.)
- Por estas razões usa-se um módulo de Entradas e Saídas que tem as seguintes funções:
  - o Fornecer uma interface com a CPU e a memória via o bus de sistema ou comutador central (central switch)
  - o Permitir a interface com um ou mais dispositivos periféricos

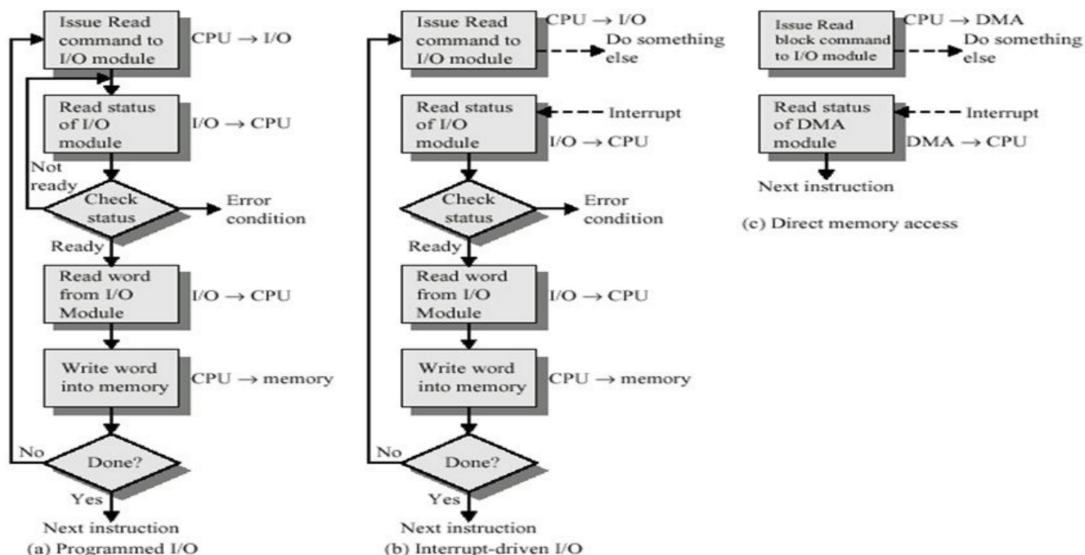


#### Dispositivos Externos de I/O:

- Um dispositivo externo é ligado ao computador através de uma ligação a um módulo de I/O
- Human-readable
  - o Comunicação com o usuário (impressoras, terminais vídeo, etc.)
- Machine-readable
  - o Comunicação com equipamento (discos, sensores, etc.)
- Communication
  - o Comunicação com dispositivos remotos (terminais, placas de redes)

#### Estrutura dos Módulos:

- O módulo de entradas e saídas, contém lógica específica para a interface de cada um dos dispositivos que ele controla
- Existem três técnicas de operação com os módulos de I/O:
  - o Entradas e saídas programadas
  - o Entradas e saídas comandadas por interrupções
  - o Direct Memory Access (DMA)



#### Entrada e Saídas Programadas

- Os dados são transferidos entre a CPU e o módulo de E/S
- A CPU executa um programa que lhe dá controle direto das entradas e saídas incluindo a monitoração do estado,

- envio de comandos de leitura e escrita e transferência de dados
- A CPU tem de esperar que a operação de entradas e saídas termine
  - Se o processador for mais rápido que I/O, existe uma perda de tempo do processador
  - Para executar uma instrução de I/O, a CPU envia um endereço (especificando um dispositivo externo) e um comando de I/O
  - Existem 4 tipos de comandos de I/O:
    - o Controle: ativa o periférico e diz o que fazer
    - o Teste: testa as várias condições de estado associadas a um módulo de I/O e aos seus periféricos
    - o Ler: faz com que o módulo obtenha dados do periférico e os coloque no buffer
    - o Escribir: faz com que o módulo leia uma palavra do bus e a envie para o periférico
  - A cada dispositivo é associado um identificador ou endereço distinto
  - Quando o processador, a memória e os módulos de I/O partilham o mesmo bus, existem dois modos de endereçamento possíveis:
    - o Memory-mapped I/O
    - o Isolated I/O
  - Uma desvantagem deste sistema tem a ver com a utilização do espaço de endereçamento valioso que é a memória

#### **I/O Comandada por Interrupções:**

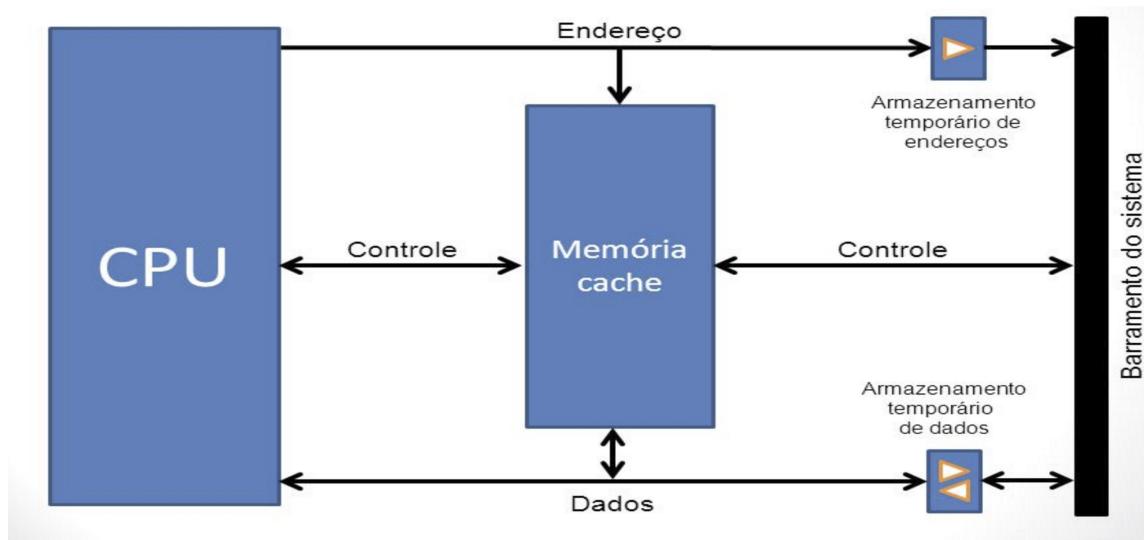
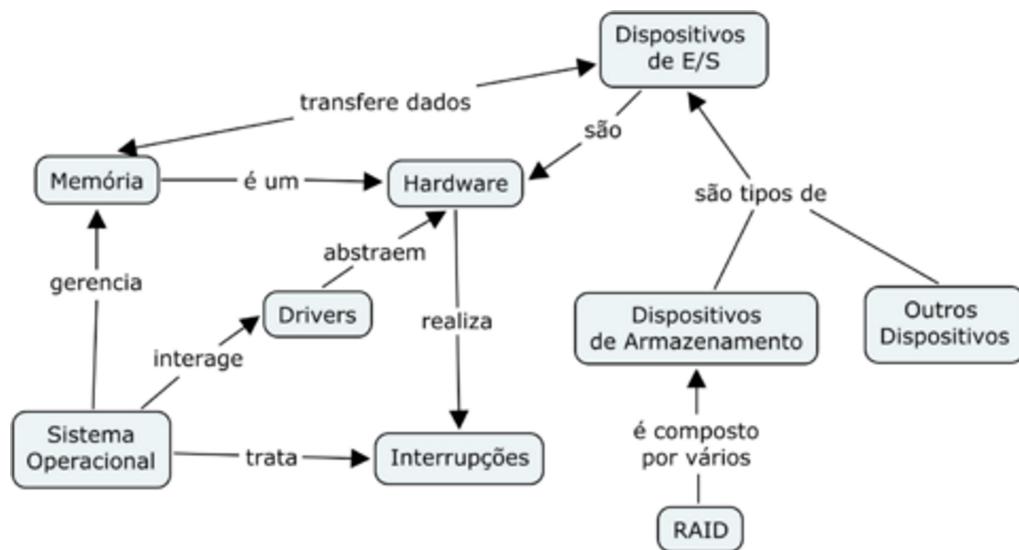
- Uma alternativa à I/O programada é a CPU enviar um comando de I/O para um módulo e continuar a execução, este interromperá então a CPU quando estiver pronto para transferir dados
- A CPU
  - o A CPU envia um comando de leitura e continuar a execução
  - o No fim de cada instrução a CPU testa as interrupções e quando detecta uma interrupção do módulo, o lê os dados do bus e guarda-os em memória
- O módulo
  - o O módulo recebe um comando de leitura da CPU e lê os dados do periférico associado
  - o Quando os dados estão no registrador da dados do módulo, este interrompe a CPU e aguarda que os dados sejam requisitados, colocando-os no bus
- O Processamento das Interrupções:
  - 1.O dispositivo envia um sinal de interrupção ao processador
  - 2.O processador acaba a instrução atual antes de responder à interrupção
  - 3.O processador testa a existência de interrupções e comunica ao dispositivo um sinal de acknowledgement ao dispositivo que enviou a interrupção, para que este retire o sinal de interrupção
  - 4.O processador prepara a transferência do controle para a rotina de interrupção
    - a. Guarda o estado do processador (que está no registrador PSW – program status word)
    - b. Guarda a localização da próxima instrução a ser executada (que está guardada no program counter)
    - c. Esta informação pode ser salvada na stack do sistema
  - 5.O processador carrega no PC a localização do interrupt handling programa que responde à interrupção
  - 6.Um novo contexto é alcançado com a execução da rotina de tratamento da interrupção
  - 7.À medida que a interrupção é processada, é efetuada uma verificação da ocorrência ou não de outras interrupções. Isto pode envolver o envio adicional de comandos para os dispositivos de I/O
  - 8.Quando a interrupção acabou de ser tratada, é necessário recuperar o contexto do sistema que tinha sido salvado na pilha (stack)
  - 9.O programa continua a sua execução a partir da instrução que tinha sido guardada no PC
- Como identificar o módulo que emitiu uma interrupção?
  - o Múltiplas Linhas de Interrupção
  - o Identificação por Software
  - o Daisy Chain
  - o Bus Master
- Como gerir várias interrupções?
  - o i.e., as ações de resposta a uma interrupção é interrompida por uma outra interrupção

#### **Acesso Direto à Memória (DMA-Direct Memory Access):**

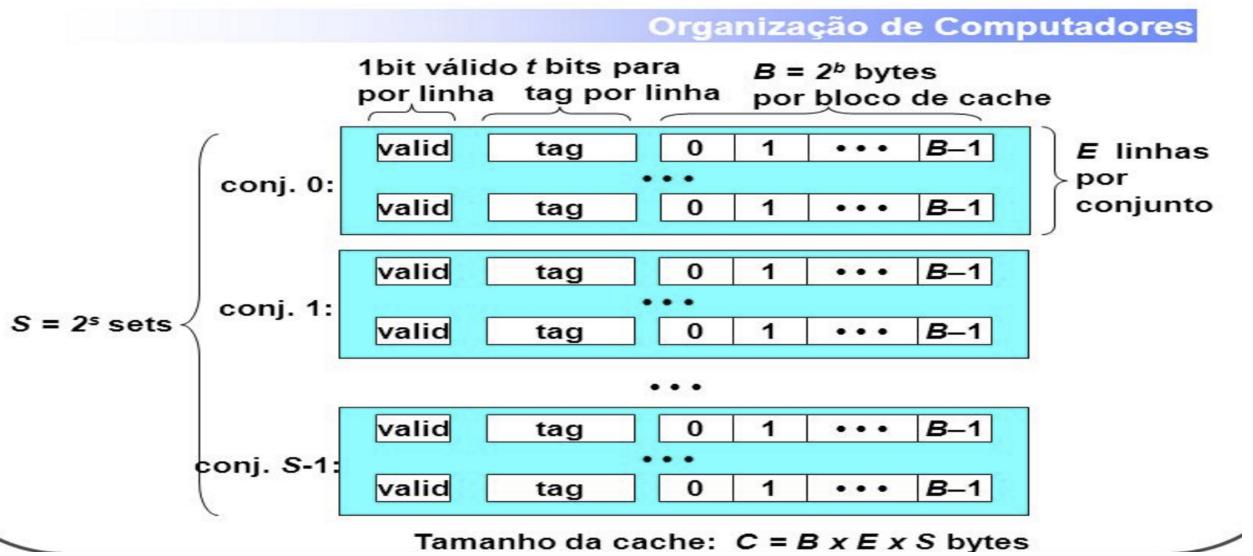
- Note-se que não se trata de uma interrupção, o processador não salva nenhum contexto apenas executa uma outra função
- O que pode acontecer é que o processador pare por um ciclo de bus, Na generalidade o processador pode executar as instruções mais lentamente
- Em sistemas de transferência de palavras de I/O múltiplos,
- DMA torna-se muito mais eficiente do que os mecanismos anteriores

- a) Todos os módulos partilham a mesma estrutura de bus. Esta configuração, embora mais barata, torna-se ineficiente pois cada transferência ocupa dois ciclos de bus (para ler e escrever uma palavra)
- b) Os módulos de I/O estão diretamente ligados ao módulo DMA
- c) Utilização de um bus de I/O. Apenas o módulo de DMA partilha o bus de sistema. A partilha de dados entre o DMA e os módulos de I/O têm lugar fora do bus de sistema
- 

## EXTRAS:



# Organização geral da memória cache



DICA: Youtube -> Crash Course Computer Science (ajuda muito em OCD)