ORGANIZAÇÃO E ARQUITETURA DE COMPUTADORES

Estrutura interna de computadores

Profª. Fabiana F F Peres

Apoio: Camile Bordini

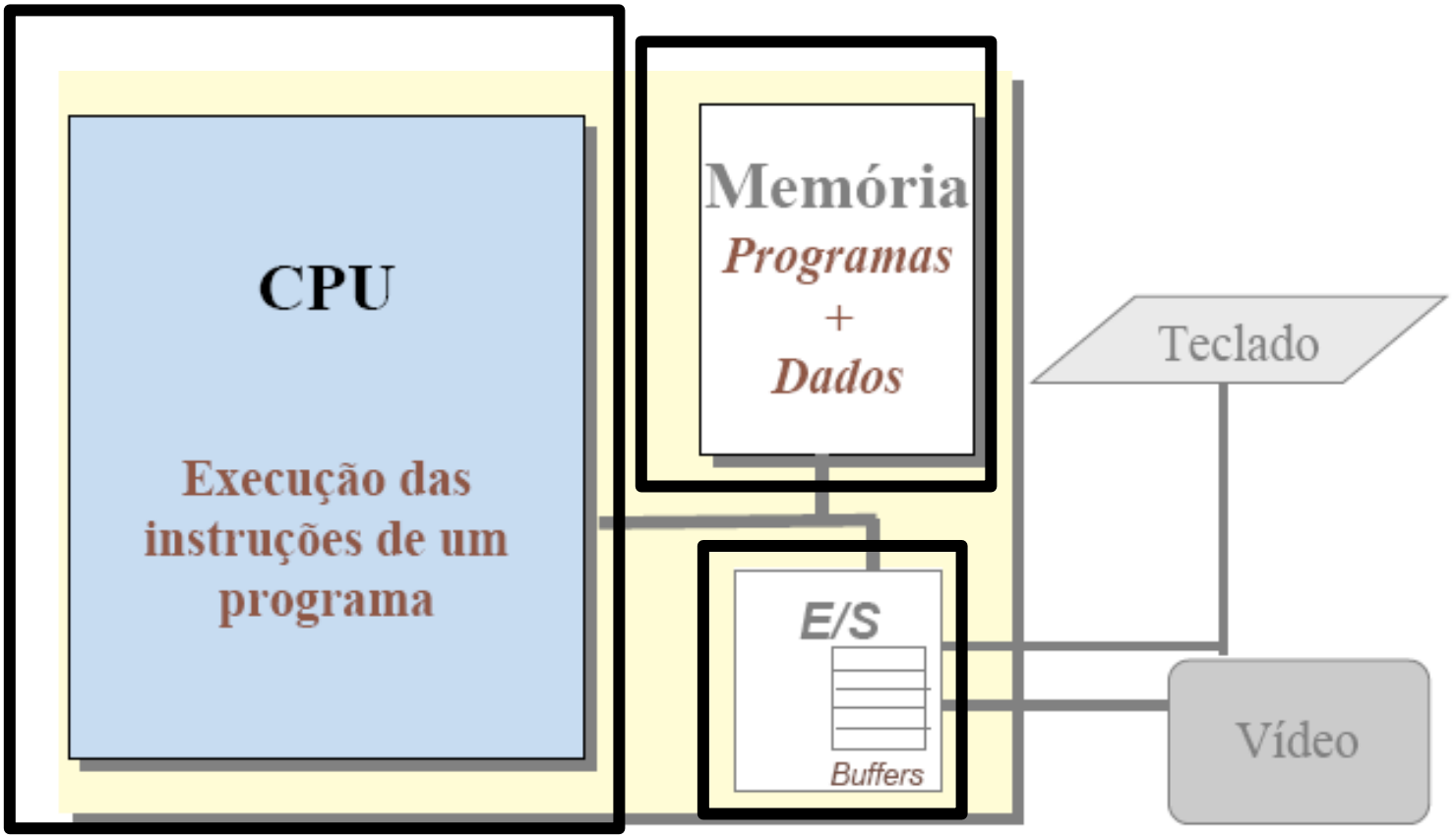
Composição Básica de um Computador

1. **Processador**

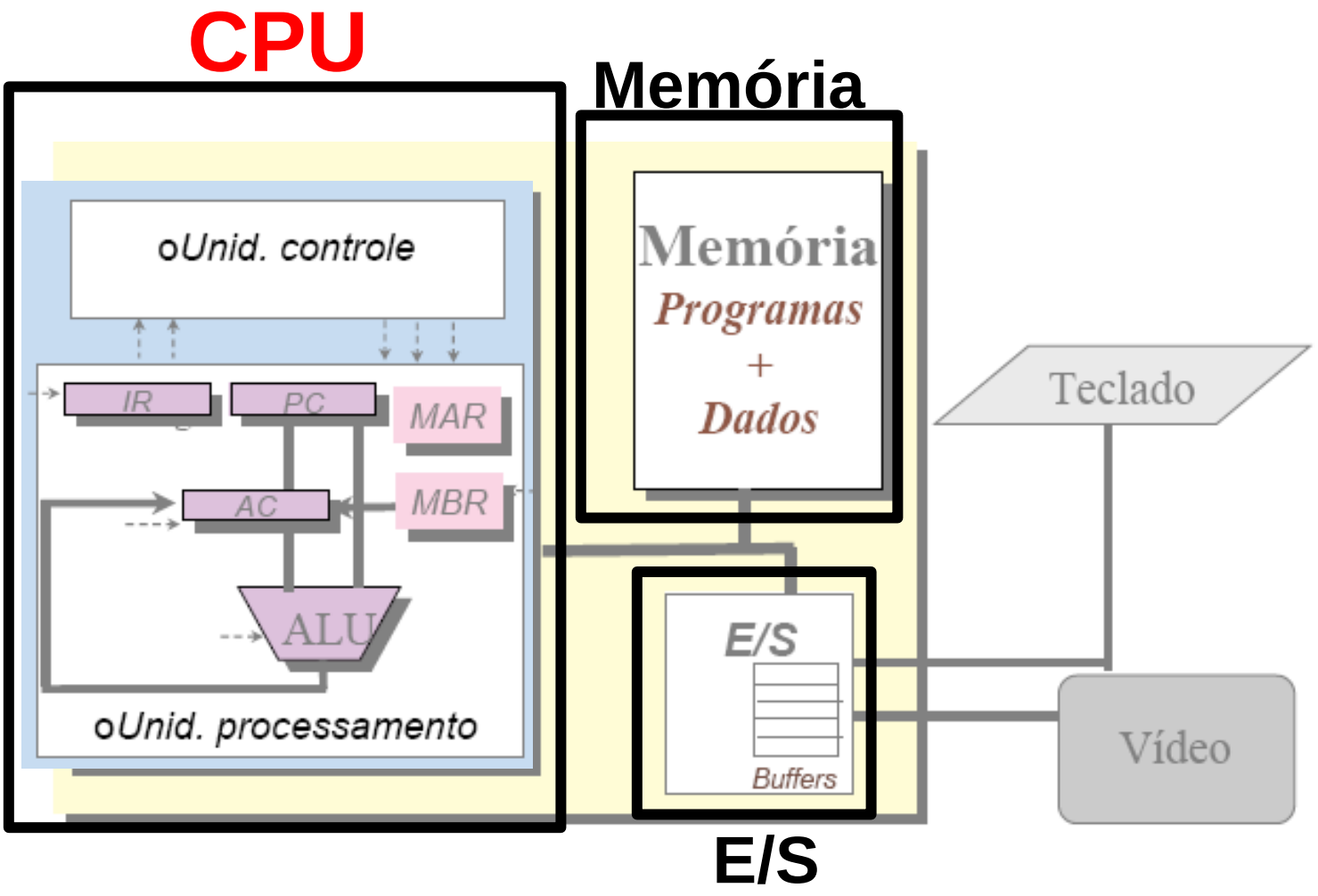
2. **Memória**
   - Memória principal
   - Memória secundária

3. **Dispositivos de entrada e saída**

Composição Básica de um Computador



1. CPU


CPU
Memória
oUnid. controle
IR
PC
MAR
AC
MBR
ALU
oUnid. processamento
Memória
Programas
+
Dados
E/S
Buffers
E/S
Teclado
Vídeo

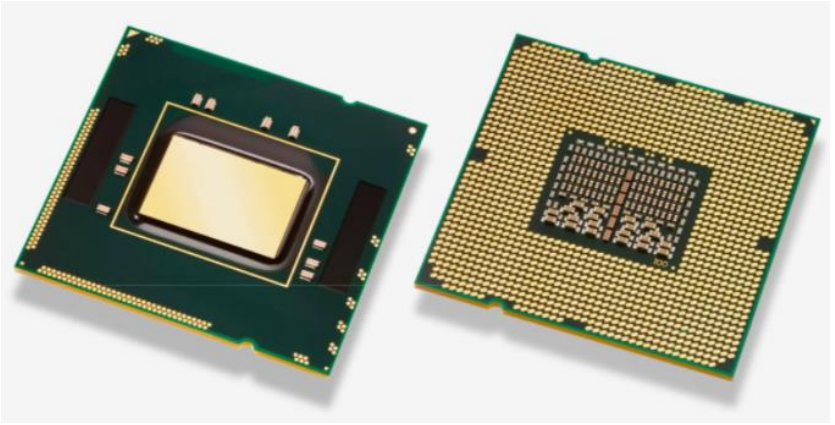
1. CPU – Principais Componentes

1.1. Unidade de Controle (UC)

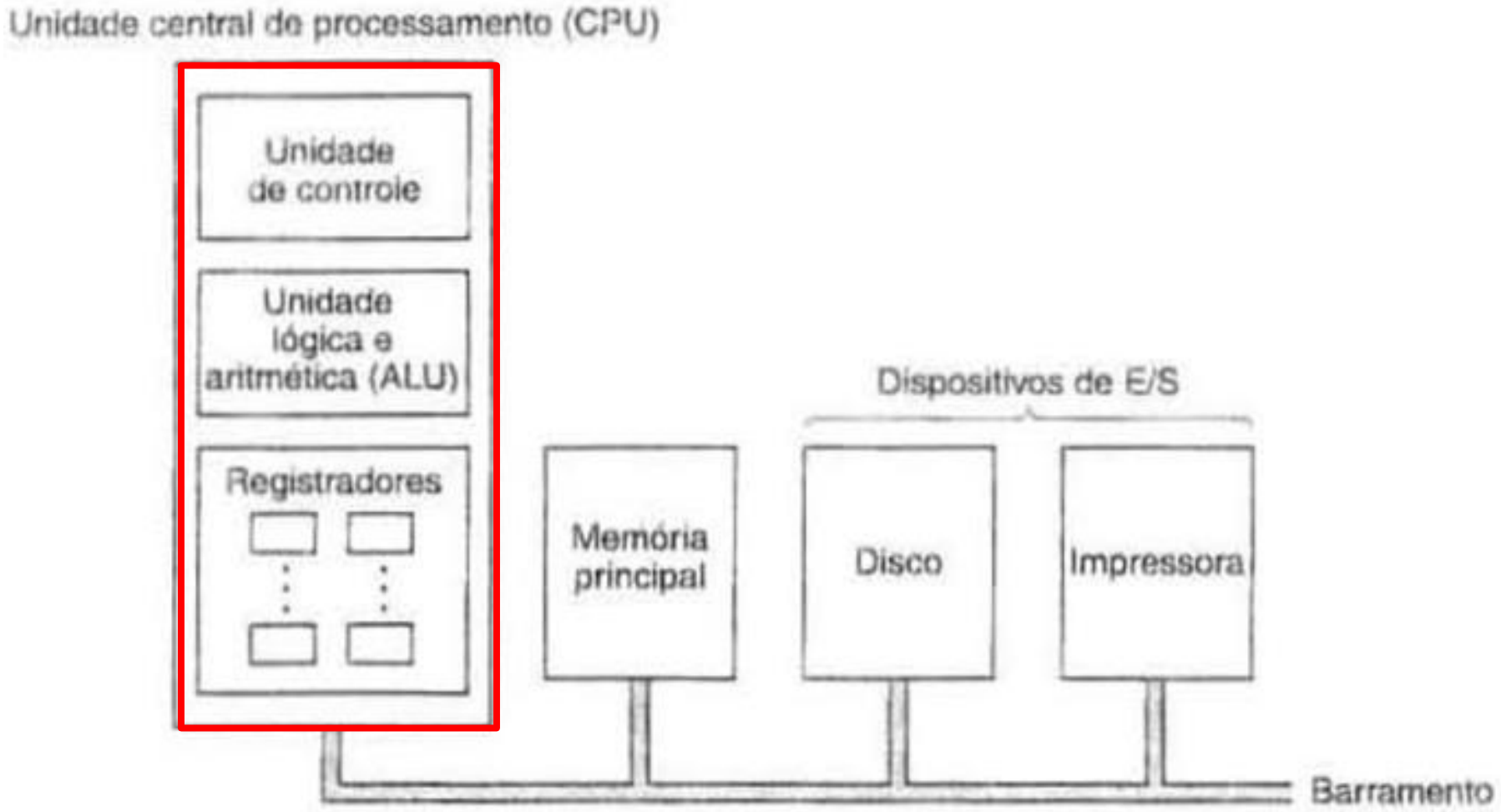
1.2. Unidade Lógica e Aritmética (ULA)

1.3. Registradores

1.4. Barramentos



1. CPU – Principais Componentes



1.1. Unidade de Controle

- Responsável por...
  - Buscar instruções da memória principal
  - Interpretar (decodificar) instruções
  - Controlar a execução de instruções
  - Controlar os demais componentes do computador, como memória e dispositivos de entrada e saída

1.1. Unidade de Controle

- Exemplo de controle de execução – instrução de soma
  - Unidade de Controle ordena que a ALU carregue dois operandos;
  - Unidade de Controle determina que ALU some os dois operandos e armazene o resultado em algum registrador;

1.2. Unidade Lógica e Aritmética
Arithmetic-logic Unit (ALU)

- Realiza as operações necessárias a execução das instruções;
- Aglomerado de **circuitos lógicos** e **componentes eletrônicos** simples que, integrados, realizam...
  - *Operações aritméticas (soma, subtração, ...)*
  - *Operações lógicas (AND, OR, XOR...).*

Obs: processadores modernos utilizam mais de uma ALU

1.3. Registradores

- Memória rápida e pequena
- Usados para...
  - Armazenamento de **dados temporários** que serão usados pela ALU e de resultados gerados por ela;
  - Armazenamento de **instruções**;
  - Registro de **informações de controle**, como o número da próxima instrução a ser executada;

1.3. Registradores

- Podem ser de uso geral ou específico

- Exemplos de registradores de uso específico:
  - **Program Counter (PC)**: armazena o **endereço** de memória da **próxima instrução**
  - **Registrador de Instruções (IR)**: armazena a **instrução que está sendo executada**

1.3. Registradores

- Em geral, os registradores têm uma largura (número de bits que podem armazenar) igual ao tamanho estabelecido pelo fabricante para a palavra (*word*);

- A quantidade e o emprego dos registradores variam bastante de modelo para modelo de CPU.

Registradores

+

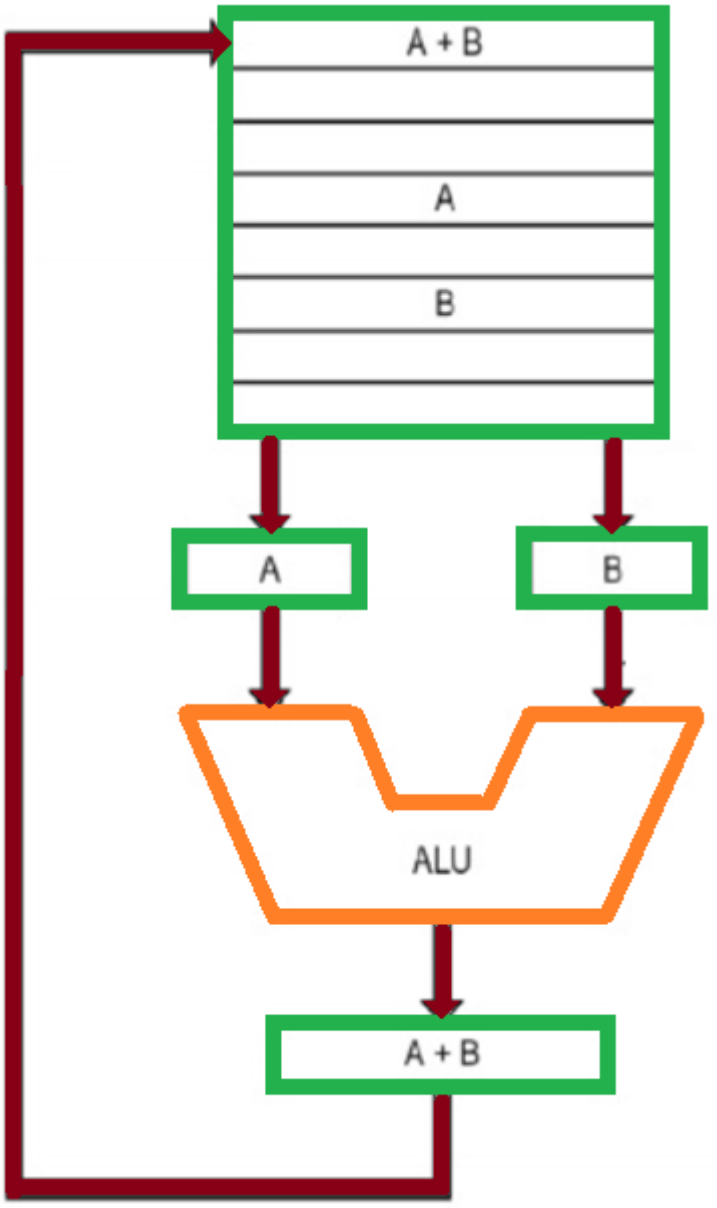
ULA

+

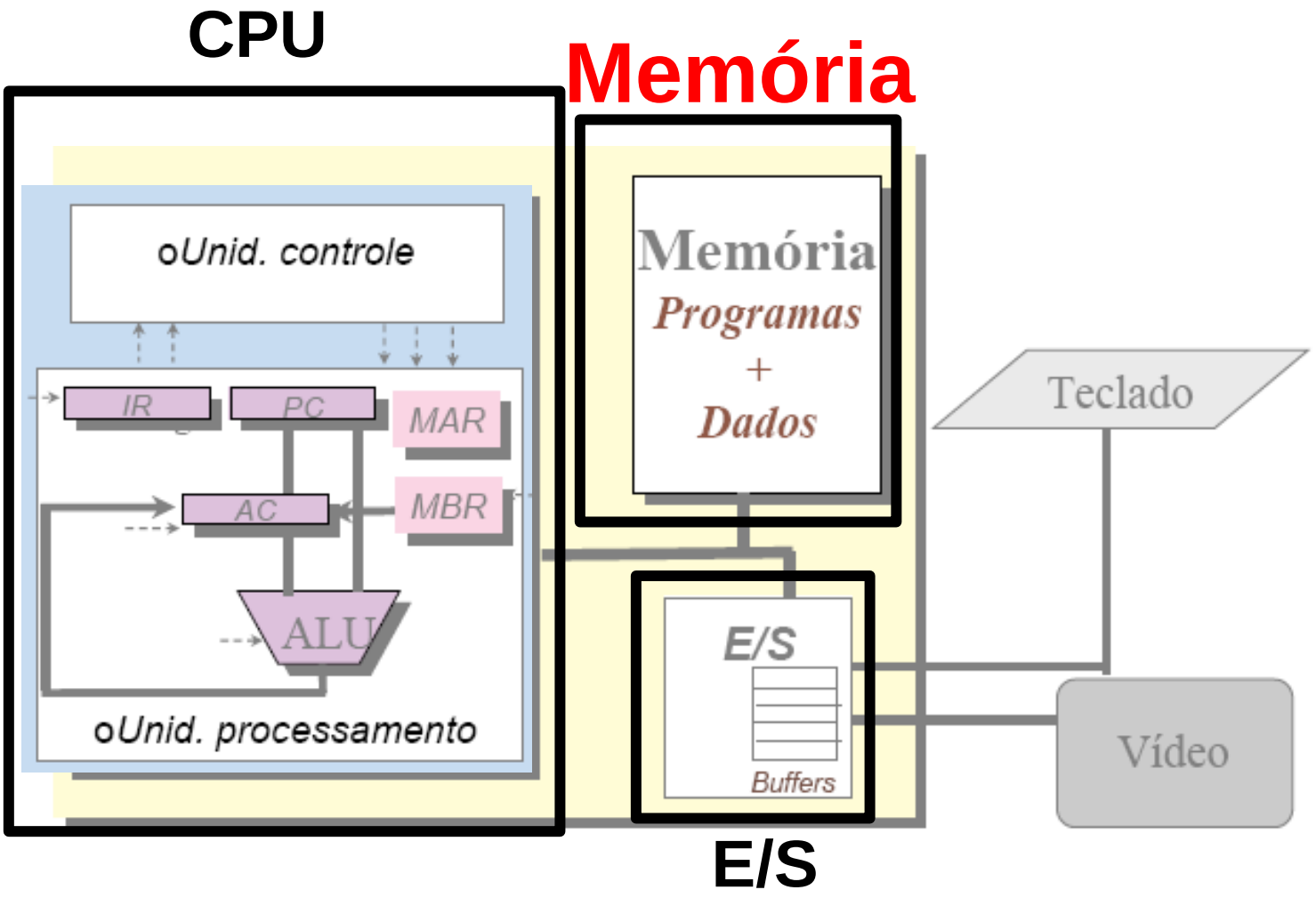
Barramentos

=

Caminho de dados
(estrutura por onde os dados fluem)



2. Memória Principal



Memória Principal - Bits

- Parte do computador onde programas e dados são armazenados

- Unidade básica: **dígito binário (bit)**

- Base de representação: **Binária**
  - Por que não é utilizado base decimal?

Memória Principal - Bits

- Um Sistema de armazenamento digital é baseado na **distinção entre valores** de alguma grandeza física (ex: corrente, tensão)

- Assim, quantos mais valores houver para serem diferenciados:
  - Menor a separação entre eles
  - Menos confiável

- Portanto, aritmética binária: mais “eficiente”

Memória Principal - Bits

- Exemplo de utilização de aritmética decimal
  - Mainframes IBM
  - Cada dígito decimal é armazenado segundo um código de **4 bits**: **BCD** (***Binary Coded Decimal***)
  - 2^4 combinações diferentes (apenas de 0 à 9 usadas)

Ex: 1944

(em **16 bits**)

	1	9	4	4
• **BCD**:	0001	1010	0100	0100

- **binário puro**: 0000011110011000

- Para o exemplo anterior (em **16 bits**), qual a faixa de representação numérica em cada caso?

- Mas, caso existisse um dispositivo eletrônico capaz de armazenar diretamente os dígitos de 0 à 9, com **4 desses dispositivos**, qual a faixa de representação numérica em cada caso?

- Para o exemplo anterior (em **16 bits**), qual a faixa de representação numérica em cada caso?
  - **BCD**: 0 à 9.999
  - **binário puro**: 0 à 65.535

- Mas, caso existisse um dispositivo eletrônico capaz de armazenar diretamente os dígitos de 0 à 9, com **4 desses dispositivos**, qual a faixa de representação numérica em cada caso?
  - **BCD**: 0 à 9.999
  - **binário puro**: 0 à 15

Endereços de Memória

- É formada por um **conjunto de células** ou **posições**
  - Cada **célula** possui um **endereço** e pode guardar uma informação
  - Uma **célula** é a menor unidade endereçável em uma memória
  - Por meio do endereço os programas podem referenciar a célula

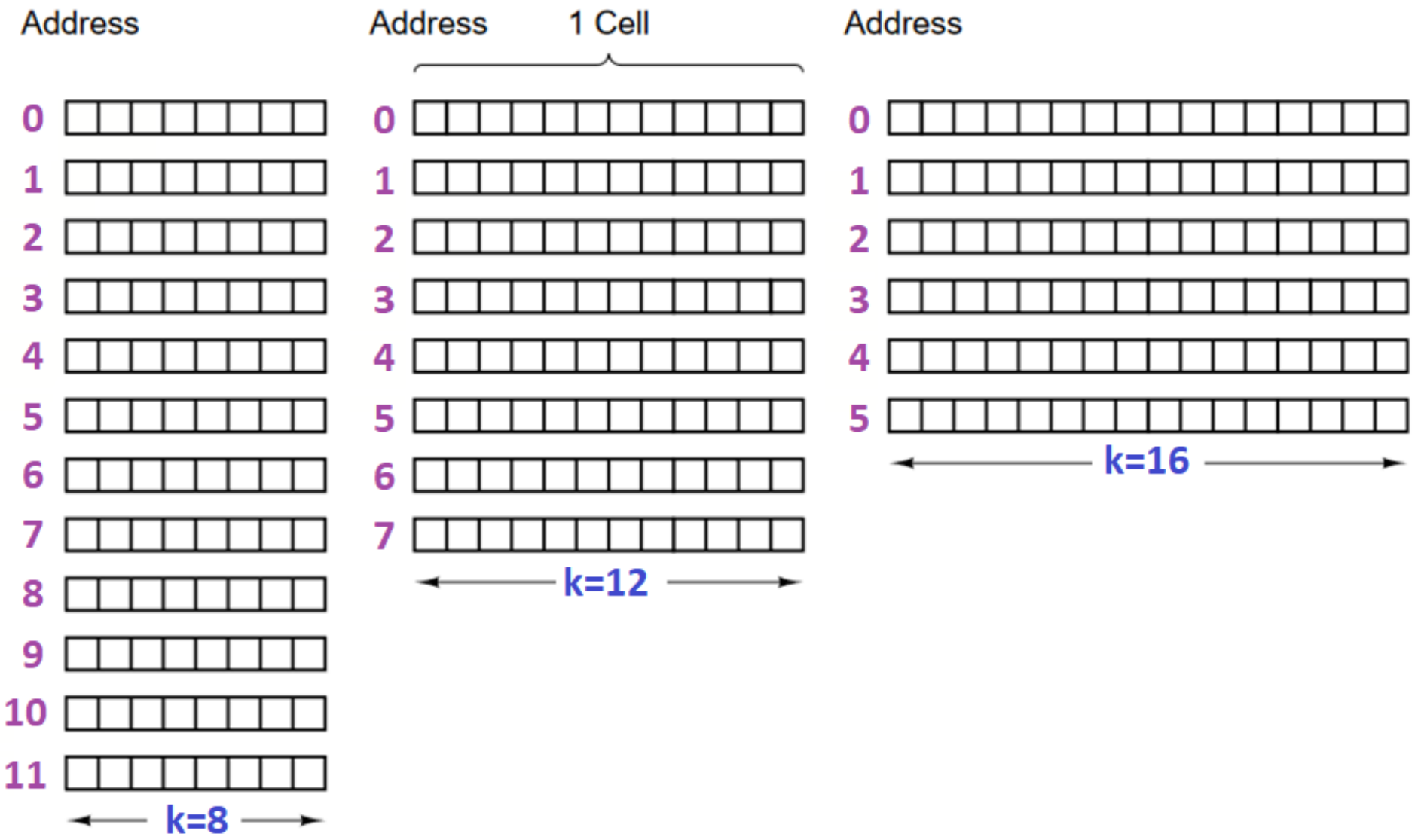
Endereços de Memória

- Em uma memória com **n** células
  - **Endereços** vão de **0** à **n-1**

- Todas as células da memória: com um mesmo número de bits
  - Com **k** bits: **2^k** combinações possíveis
  - Tamanho comum: **k=8 bits (1 byte)**

Computer	Bits/cell
Burroughs B1700	1
IBM PC	8
DEC PDP-8	12
IBM 1130	16
DEC PDP-15	18
XDS 940	24
Electrologica X8	27
XDS Sigma 9	32
Honeywell 6180	36
CDC 3600	48
CDC Cyber	60

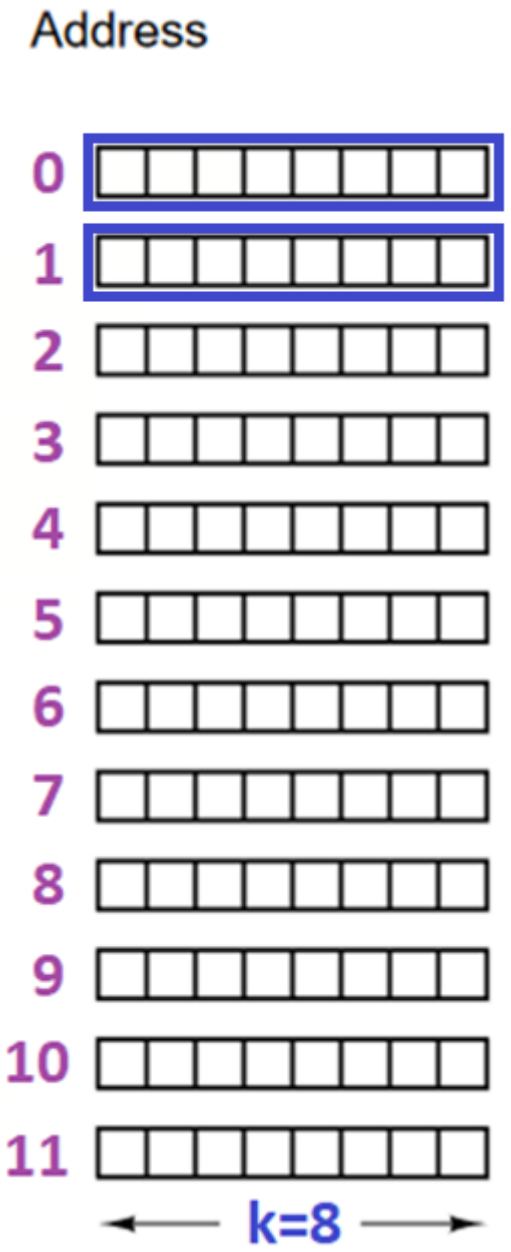
- 3 formas de se organizar uma memória de 96 bits



Endereços de Memória

- **Endereços** das célula: também expresso em binário

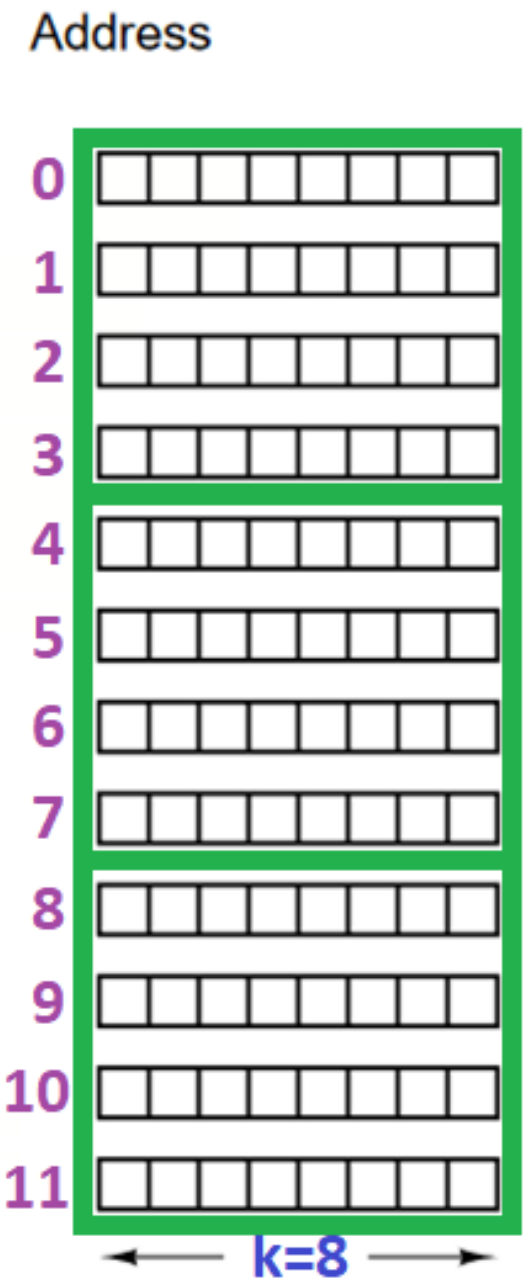
- Quantos bits são necessários para expressar todos os endereços ao lado?
  - Ex: endereços com **m bits** - máximo de 2^m células endereçáveis



Endereços de Memória

- Os **bytes** são agrupados em **palavras** (***words***)

A maioria das instruções de uma máquina opera sobre **palavras**.



- Um computador com:
  - palavra de 32 bits possui 4 bytes/palavra
  - palavra de 64 bits possui 8 bytes/palavra

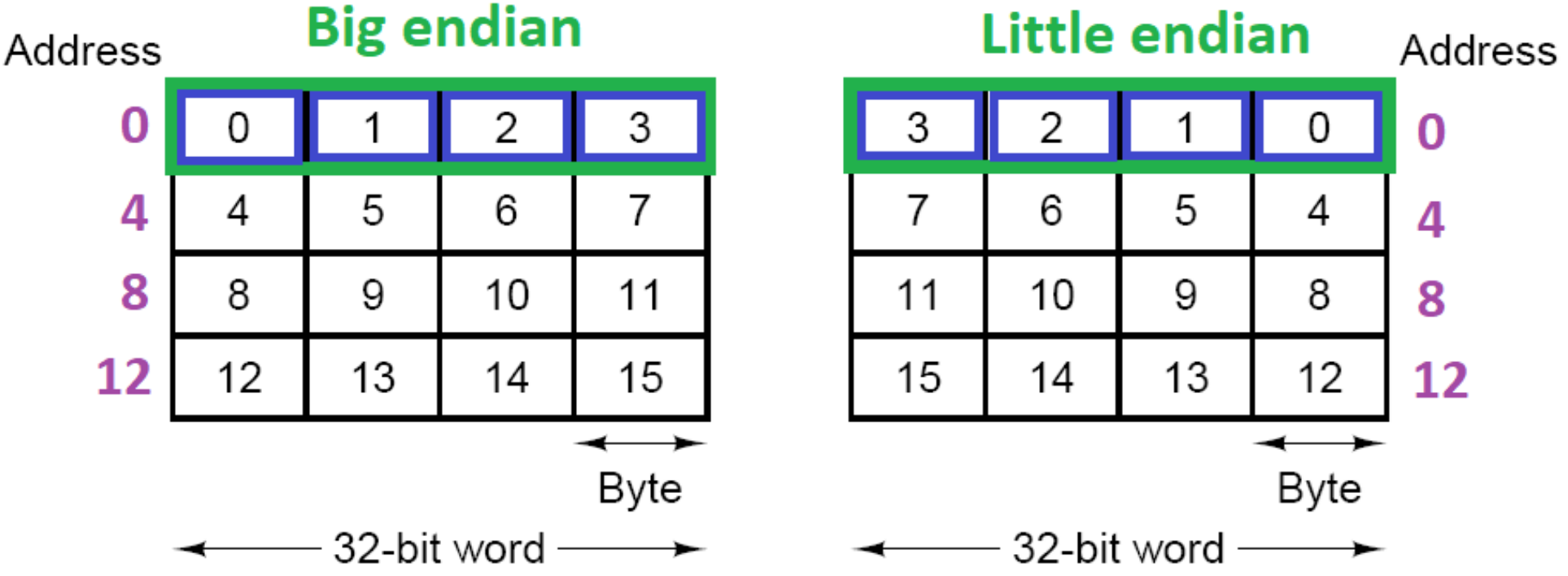
Memória - Ordenação dos Bytes

- Existem, basicamente, **2 formas de organização dos bytes em uma palavra de memória**:
    A. **Ordenação Big endian** (maior valor em primeiro lugar – *menor endereço*)
    B. **Ordenação Little endian** (menor valor em primeiro lugar)

Memória - Origem dos termos relacionados com a ordem dos bytes

Estes termos foram inseridos no jargão da computação por um artigo publicado em 1981, citando o problema e relacionando-o a um episódio mencionado no livro **As Viagens de Gulliver** – povo que foi à guerra para decidir qual a melhor maneira de quebrar ovos, se pelo maior (big) lado ou se pelo menor (little) lado.

- **Big endian: Bytes** são numerados da **esquerda para a direita** (0, 1, 2,..., n-1)
  - Ex: arquiteturas SPARC, IBM Mainframe, …
- **Little endian: Bytes** são numerados da **direita para esquerda** (n-1, ..., 2, 1, 0)
  - Ex: arquiteturas INTEL



Exemplo – Ordem dos Bytes

- Considere o seguinte registro:
  - Nome do empregado: Jim Smith
  - Idade do empregado: 21 anos
  - Departamento do empregado: 260 (1 -00000100)

- Como será a representação das informações em cada forma de organização?

Exemplo – Ordem dos Bytes

Big endian

0	J	I	M	
4	S	M	I	T
8	H	0	0	0
12	0	0	0	21
16	0	0	1	4

(a)

Little endian

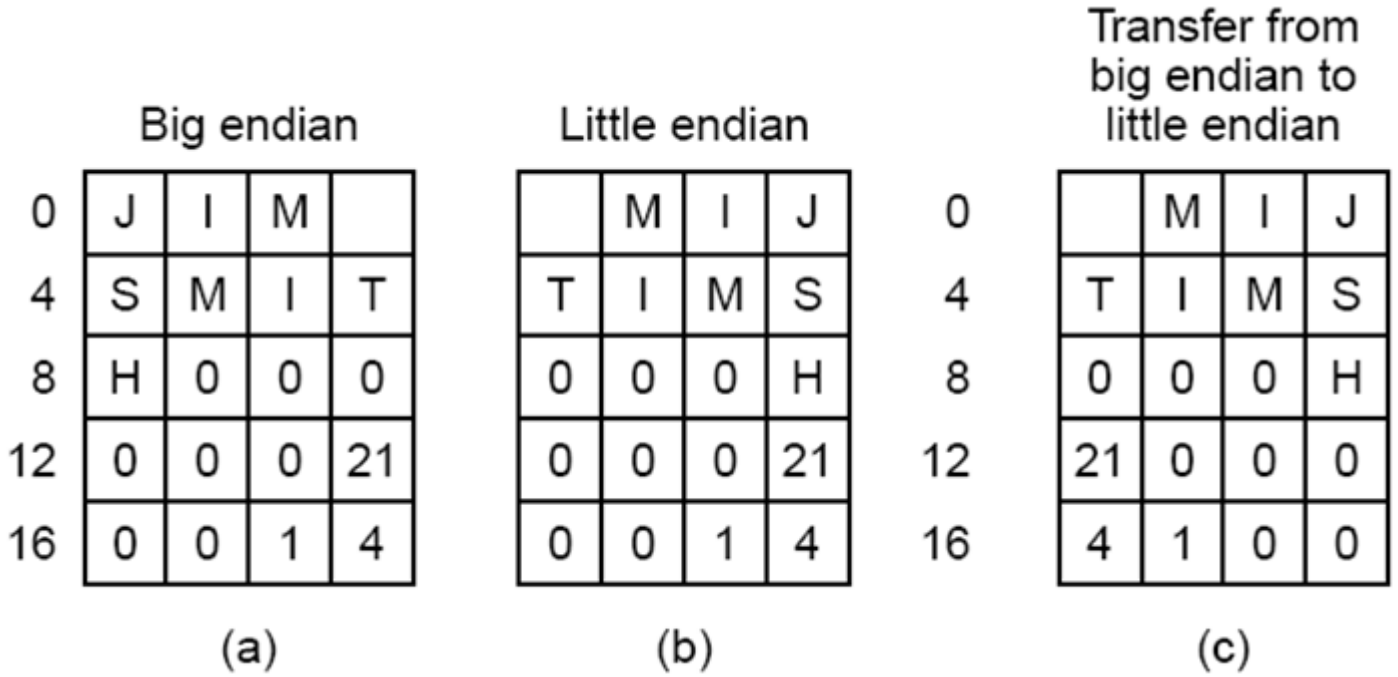
	M	I	J
T	I	M	S
0	0	0	H
0	0	0	21
0	0	1	4

(b)

Ambas as representações funcionam bem!

Problemas

- Os problemas começam quando uma máquina tenta enviar registros à outra por meio de uma rede!
- Ex: byte 0 (**Big**) -> byte 0 (**Little**) etc….. (c)

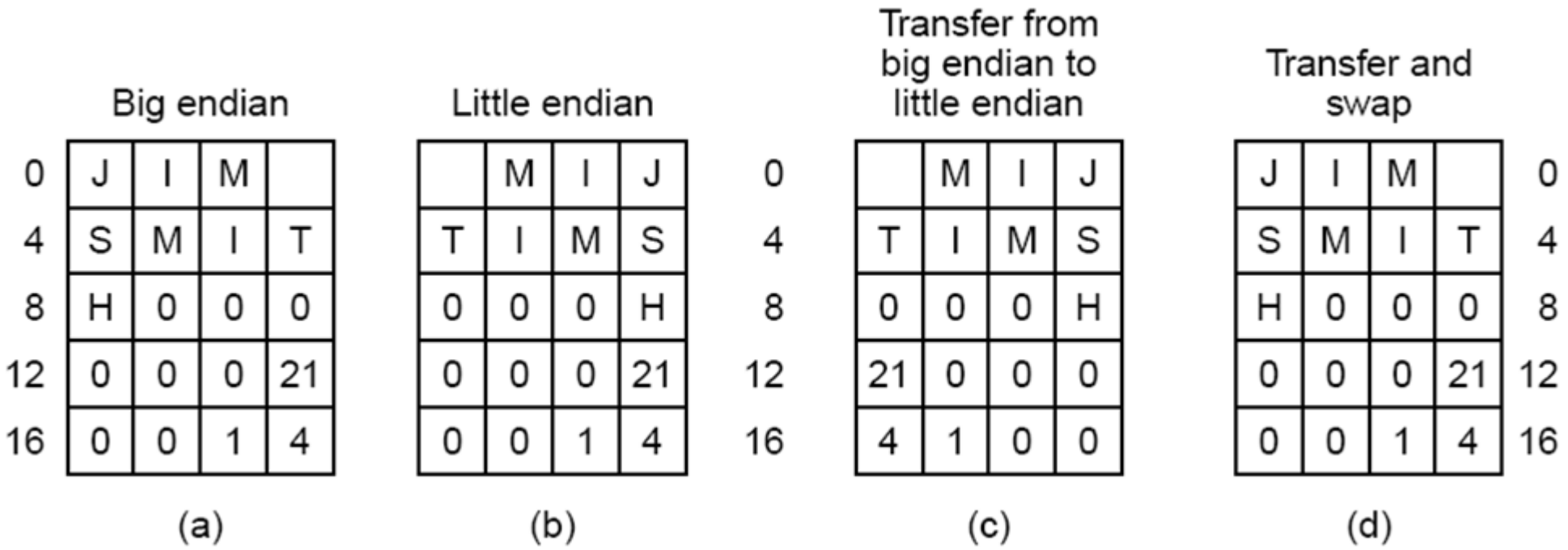


Nome: ok

Idade: 21 x 2^{24}

Problemas

- Uma solução óbvia seria um software **inverter** todos os bytes da palavra após a cópia (d)



- Porém, o problema inverte-se!

Nome: _MIJ….

Idade: ok

Problemas

- Não existe uma solução simples para este problema!

- Se incluirmos um cabeçalho junto a cada item de dados indicando o tipo (caractere, número inteiro, etc.) e o tamanho do dado
  - Funciona, mas ineficiente

- Endiannes em microcontroladores:

  https://embarcados.com.br/endiannes-em-microcontroladores/

Referências Bibliograficas

Tanenbaum, A S “**Organização Estruturada de Computadores”** – Prentice Hall do Brasil 5ª edição, 2006; capitulo 2(Tanenbaum): 2.1→ 2.1.1, 2.1.2 (pg 29.); 2.2 → 2.2.1, 2.2.2, 2.2.3 e 2.2.4

Stallings, William; “**Computer Organization and Architecture – Designing for Performance**”. 8° ed. Prentice Hall, Inc., New Jersey, 2010; **appendix 10: pg 396 (ordem dos bytes).**