Arquitectura de Computadores I

Organização da memória

Miguel Barão

ESCOLA DE CIÊNCIAS E TECNOLOGIA
DEPARTAMENTO DE INFORMÁTICA

Resumo

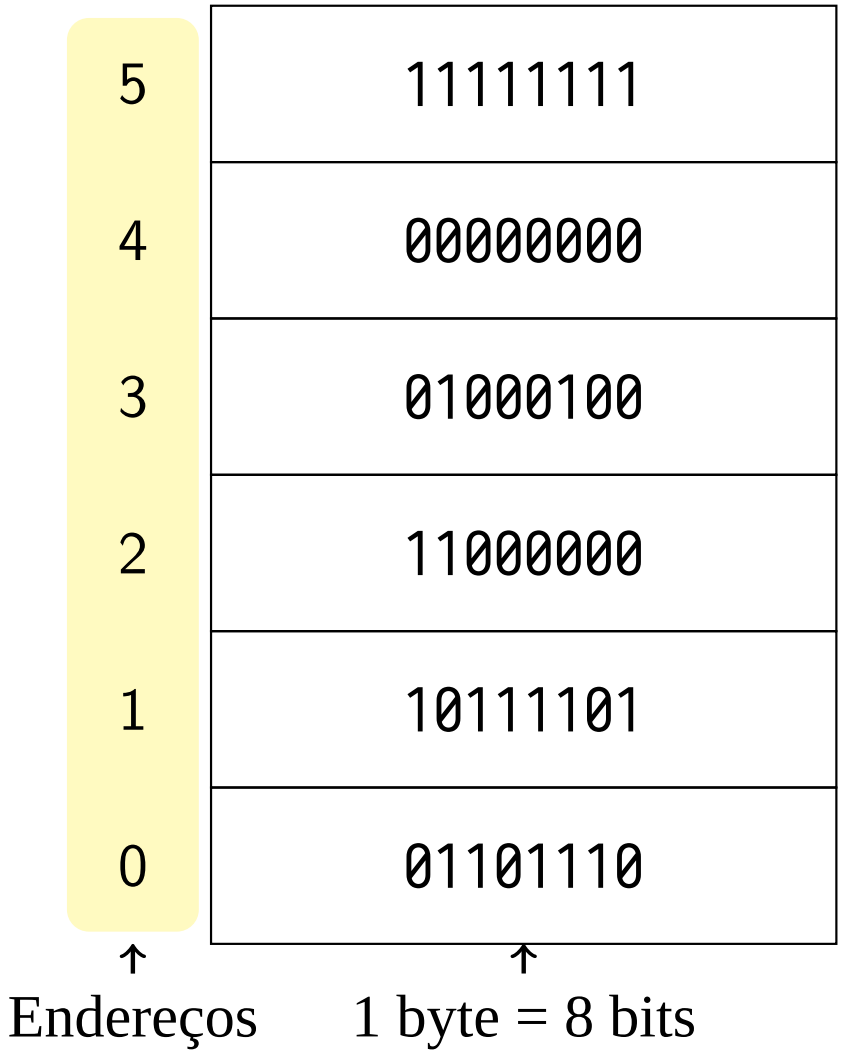
Organização da memória

Ordenação de bytes (*endianness*)

Prefixos binários e decimais

Organização da memória

Memória: organização em bytes



Funciona como uma tabela onde as linhas são numeradas e cada uma armazena exactamente 8 bits de informação (1 byte = 1 octeto).

Unidades de informação

1 bit é abreviatura de "binary unit".
Pode tomar os valores 0 ou 1.
Os sistemas digitais funcionam na base binária.

1 byte é a quantidade de informação que ocupa cada endereço de memória.
Normalmente são 8 bits.

1 word é a unidade de informação natural de uma arquitectura.
As instruções processam dados deste tamanho.
Na arquitectura RISC-V, 1 word = 32 bits, mas outras arquitecturas podem ter outros tamanhos (16 bits, 64 bits, etc).

bit $\leftrightarrow$ matemática, byte $\leftrightarrow$ memória, word $\leftrightarrow$ processador

Memória: endereçamento

Atenção

Não é possível endereçar directamente um bit em memória.
Um endereço apenas permite aceder a 1 byte (bloco de 8 bits).

As operações permitidas sobre a memória são:

- Ler um byte da memória (*load byte*)
- Escrever um byte na memória (*store byte*)

Também é possível escrever vários bytes de uma vez.
Por exemplo:

- Ler uma word da memória (*load word*)
- Escrever uma word na memória (*store word*)

Exemplo

Suponha que a memória de um computador contém a seguinte informação:

Endereço	Byte
3	10000000
2	11111111
1	00000000
0	00000111

1. Qual o conteúdo da memória nos endereços 0, 1, 2 e 3?
2. Represente em decimal e hexadecimal os bytes nesses endereços.
3. Se os bytes representam números inteiros em complemento para 2, que números são?

Representação de números nas bases binária e hexadecimal

Para distinguir as bases de numeração usamos prefixos:

- Hexadecimais com prefixo `0x`.
- Binários com prefixo `0b`.
- Decimais não têm prefixo. Primeiro algarismo de `1` a `9`.

Exemplo

Decimal	Hexadecimal	Binário
1	0x01	0b00000001
15	0x0f	0b00001111
16	0x10	0b00010000
128	0x80	0b10000000
255	0xff	0b11111111

Representação de números nas bases binária e hexadecimal

Para distinguir as bases de numeração usamos prefixos:

- Hexadecimais com prefixo `0x`.
- Binários com prefixo `0b`.
- Decimais não têm prefixo. Primeiro algarismo de `1` a `9`.

Exemplo

Decimal	Hexadecimal	Binário
1	0x01	0b00000001
15	0x0f	0b00001111
16	0x10	0b00010000
128	0x80	0b10000000
255	0xff	0b11111111

Atenção

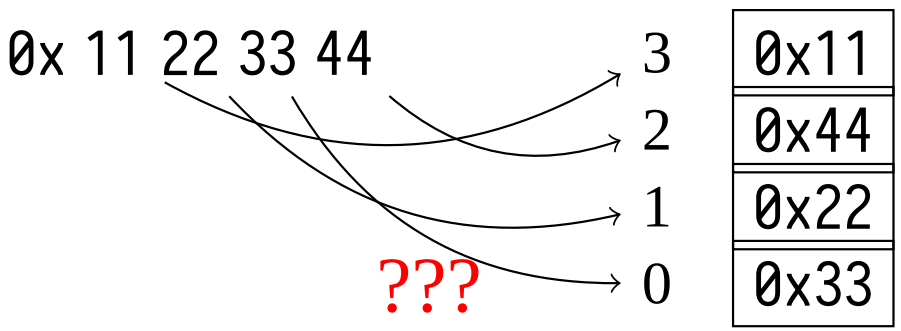
No computador todos os números estão em binário. Usa-se decimal e hexadecimal apenas por uma questão de legibilidade.

Ordenação de bytes (*endianness*)

Memória: ordenação de bytes

Ao escrever um número com mais de um byte em memória surge um problema:

- Qual a ordenação dos bytes a usar? *i.e.*,[1] deve colocar-se primeiro os bytes mais significativos ou menos significativos?



[1] *i.e.* = *id est* = "ou seja/isto é"; *e.g.* = *exampli gratia* = "por exemplo".

Ordenação de bytes (Endianness)

Definição (Endianness)

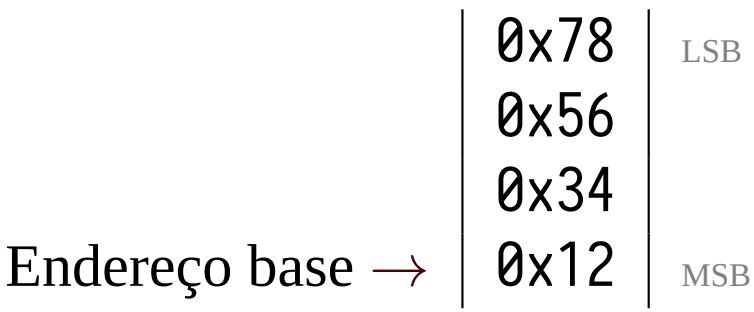
Existem duas convenções em uso:

Little endian o byte menos significativo primeiro.

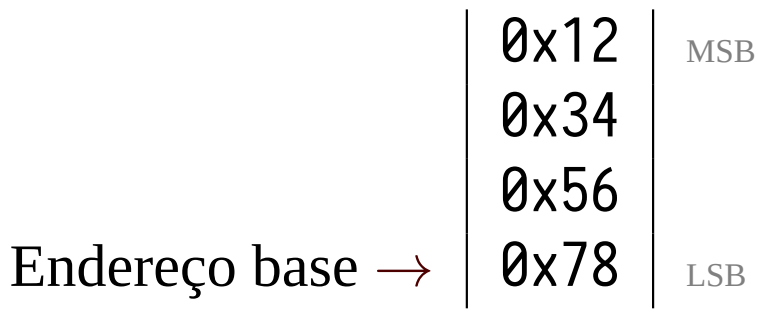
Big endian o byte mais significativo primeiro.

Memória: ordenação de bytes

Como escrever o número `0x12345678` em memória?



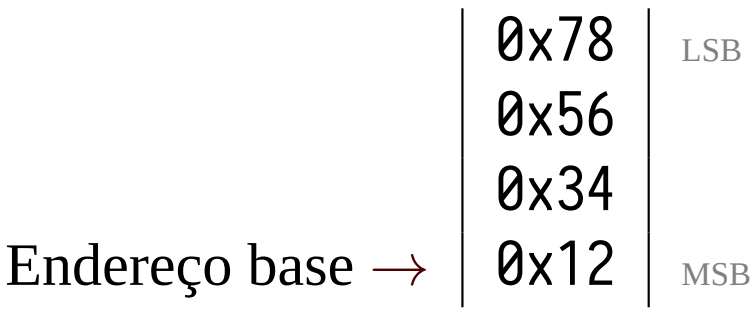
?



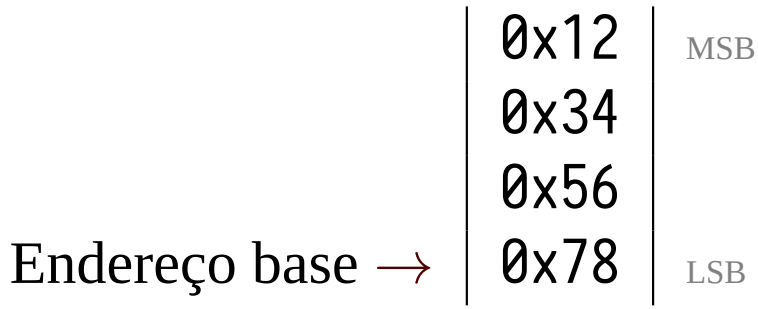
?

Memória: ordenação de bytes

Como escrever o número `0x12345678` em memória?



Big Endian



Little Endian

Memória: ordenação de bytes

A definição e o tratamento correcto da ordenação de bytes é especialmente importante quando é necessário transmitir informação entre máquinas com ordenações diferentes:

- ler/escrever um ficheiro (música, imagem, etc).
- enviar informação pela rede.

Memória: ordenação de bytes

A definição e o tratamento correcto da ordenação de bytes é especialmente importante quando é necessário transmitir informação entre máquinas com ordenações diferentes:

- ler/escrever um ficheiro (música, imagem, etc).
- enviar informação pela rede.

Arquitecturas diferentes usam ordenações diferentes:

- *Little endian* é usado nas arquitecturas x86, x86-64 e RISC-V.
- *Big endian* é usado nas arquitecturas PowerPC e SPARC.
- Algumas arquitecturas suportam ambos os modos. É o caso das arquitecturas MIPS e ARM. São usados os termos MIPSEL e ARMEL para realçar que a ordenação de bytes usada é little endian.

Exemplo

```c
#include <stdio.h>
int main() {
    int x[] = {0x12345678, 0xaabbccdd};
    FILE *fdesc = fopen("ficheiro", "w");
    fwrite(x, sizeof(int), 2, fdesc);
    fclose(fdesc);
    return 0;
}
```

Exemplo

```c
#include <stdio.h>
int main() {
    int x[] = {0x12345678, 0xaabbccdd};
    FILE *fdesc = fopen("ficheiro", "w");
    fwrite(x, sizeof(int), 2, fdesc);
    fclose(fdesc);
    return 0;
}
```

Compilando e executando, obtemos

```
$ cc a.c && ./a.out
$ hexdump -C ficheiro
00000000  78 56 34 12 dd cc bb aa
00000008
```

Qual a ordenação: *little* ou *big endian*?

Exemplo

```c
#include <stdio.h>
int main() {
    int x[] = {0x12345678, 0xaabbccdd};
    FILE *fdesc = fopen("ficheiro", "w");
    fwrite(x, sizeof(int), 2, fdesc);
    fclose(fdesc);
    return 0;
}
```

Compilando e executando, obtemos

```
$ cc a.c && ./a.out
$ hexdump -C ficheiro
00000000  78 56 34 12 dd cc bb aa
00000008
```

Qual a ordenação: *little* ou *big endian*? Little endian

Prefixos binários e decimais

Prefixos binários e decimais

Quando as grandezas são muito grandes usam-se prefixos:

SI			IEC[2]		
$10^3 = 1000$	k	kilo	$2^{10} = 1024$	Ki	kibi
10^6	M	mega	2^{20}	Mi	mebi
10^9	G	giga	2^{30}	Gi	gibi
10^{12}	T	tera	2^{40}	Ti	tebi
10^{15}	P	peta	2^{50}	Pi	pebi
10^{18}	E	exa	2^{60}	Ei	exbi
10^{21}	Z	zetta	2^{70}	Zi	zebi
10^{24}	Y	yotta	2^{80}	Yi	yobi

Pergunta: Uma memória com 16 GiB tem quantos bytes?

[2] Standard ISO/IEC 80000

Prefixos binários e decimais

Quando as grandezas são muito grandes usam-se prefixos:

SI			IEC[2]		
$10^3 = 1000$	k	kilo	$2^{10} = 1024$	Ki	kibi
10^6	M	mega	2^{20}	Mi	mebi
10^9	G	giga	2^{30}	Gi	gibi
10^{12}	T	tera	2^{40}	Ti	tebi
10^{15}	P	peta	2^{50}	Pi	pebi
10^{18}	E	exa	2^{60}	Ei	exbi
10^{21}	Z	zetta	2^{70}	Zi	zebi
10^{24}	Y	yotta	2^{80}	Yi	yobi

Pergunta: Uma memória com 16 GiB tem quantos bytes?

$16\ \text{GiB} = 16 \times 2^{30}\ \text{B} =$

[2] Standard ISO/IEC 80000

Prefixos binários e decimais

Quando as grandezas são muito grandes usam-se prefixos:

SI			IEC[2]		
$10^3 = 1000$	k	kilo	$2^{10} = 1024$	Ki	kibi
10^6	M	mega	2^{20}	Mi	mebi
10^9	G	giga	2^{30}	Gi	gibi
10^{12}	T	tera	2^{40}	Ti	tebi
10^{15}	P	peta	2^{50}	Pi	pebi
10^{18}	E	exa	2^{60}	Ei	exbi
10^{21}	Z	zetta	2^{70}	Zi	zebi
10^{24}	Y	yotta	2^{80}	Yi	yobi

Pergunta: Uma memória com 16 GiB tem quantos bytes?

$16\ \text{GiB} = 16 \times 2^{30}\ \text{B} = 2^4 \times 2^{30}\ \text{B} =$

[2] Standard ISO/IEC 80000

Prefixos binários e decimais

Quando as grandezas são muito grandes usam-se prefixos:

SI			IEC[2]		
$10^3 = 1000$	k	kilo	$2^{10} = 1024$	Ki	kibi
10^6	M	mega	2^{20}	Mi	mebi
10^9	G	giga	2^{30}	Gi	gibi
10^{12}	T	tera	2^{40}	Ti	tebi
10^{15}	P	peta	2^{50}	Pi	pebi
10^{18}	E	exa	2^{60}	Ei	exbi
10^{21}	Z	zetta	2^{70}	Zi	zebi
10^{24}	Y	yotta	2^{80}	Yi	yobi

Pergunta: Uma memória com 16 GiB tem quantos bytes?

$16\ \text{GiB} = 16 \times 2^{30}\ \text{B} = 2^4 \times 2^{30}\ \text{B} = 2^{34}$ bytes.

[2] Standard ISO/IEC 80000

Utilização dos prefixos binário e decimal

- A utilização do standard IEC é relativamente recente (2008), pelo que é necessário alguma cautela.

Utilização dos prefixos binário e decimal

- A utilização do standard IEC é relativamente recente (2008), pelo que é necessário alguma cautela.

Alguns exemplos:

RAM sempre prefixos binários IEC (*e.g.* 16 GiB).

Utilização dos prefixos binário e decimal

- A utilização do standard IEC é relativamente recente (2008), pelo que é necessário alguma cautela.

Alguns exemplos:

RAM	sempre prefixos binários IEC (*e.g.* 16 GiB).
SSD, HDD	sempre prefixos decimais (*e.g.* 500 GB), mas programas podem apresentar tamanho com prefixos diferentes...

Utilização dos prefixos binário e decimal

- A utilização do standard IEC é relativamente recente (2008), pelo que é necessário alguma cautela.

Alguns exemplos:

RAM	sempre prefixos binários IEC (*e.g.* 16 GiB).
SSD, HDD	sempre prefixos decimais (*e.g.* 500 GB), mas programas podem apresentar tamanho com prefixos diferentes...
DVD	prefixo decimal.
CD	prefixo binário porque é anterior ao standard.

Utilização dos prefixos binário e decimal

- A utilização do standard IEC é relativamente recente (2008), pelo que é necessário alguma cautela.

Alguns exemplos:

RAM	sempre prefixos binários IEC (*e.g.* 16 GiB).
SSD, HDD	sempre prefixos decimais (*e.g.* 500 GB), mas programas podem apresentar tamanho com prefixos diferentes...
DVD	prefixo decimal.
CD	prefixo binário porque é anterior ao standard.
Ficheiros	tamanho reportado depende do S.O. e programas usados.

Utilização dos prefixos binário e decimal

- A utilização do standard IEC é relativamente recente (2008), pelo que é necessário alguma cautela.

Alguns exemplos:

RAM	sempre prefixos binários IEC (*e.g.* 16 GiB).
SSD, HDD	sempre prefixos decimais (*e.g.* 500 GB), mas programas podem apresentar tamanho com prefixos diferentes...
DVD	prefixo decimal.
CD	prefixo binário porque é anterior ao standard.
Ficheiros	tamanho reportado depende do S.O. e programas usados.

Utilização correcta

Prefixo binário para a memória, decimal para tudo o resto.

Exercícios

Problema

1. *Considere os números de 32 bits:* `127`, `77`, `1024`.
   - 1.1 *Escreva-os em binário e em hexadecimal.*
   - 1.2 *Represente-os de modo a ocuparem quatro endereços de memória consecutivos.*
   - 1.3 *Repita para o simétrico de cada um dos números.*
2. *Quatro endereços de memória consecutivos contêm* `0x01`, `0x7f`, `0xfc`, `0x10`, *respectivamente. Sabendo que a ordenação de bytes é Little Endian e que os quatro bytes representam um número inteiro de 32 bits, escreva o número correspondente em hexadecimal.*
3. *Quantos bits contém um ficheiro de 4 MiB?*