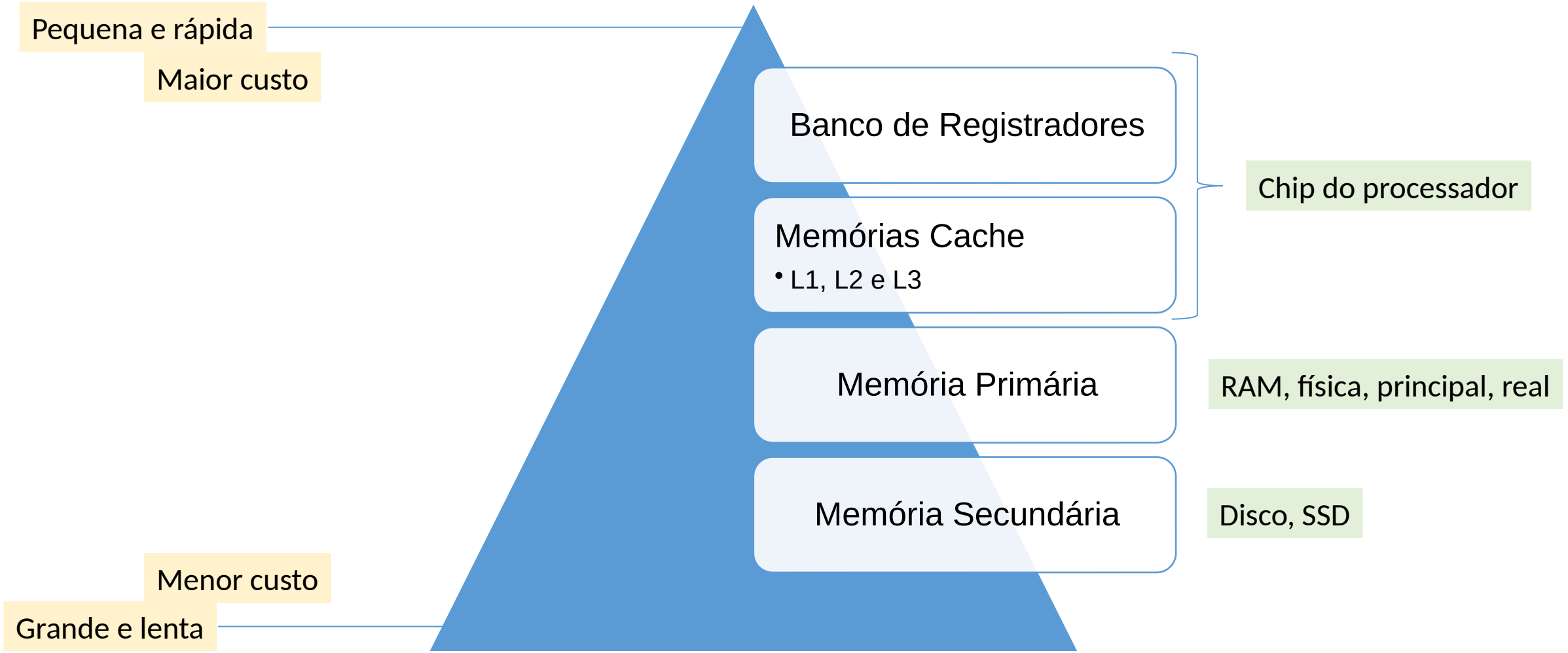
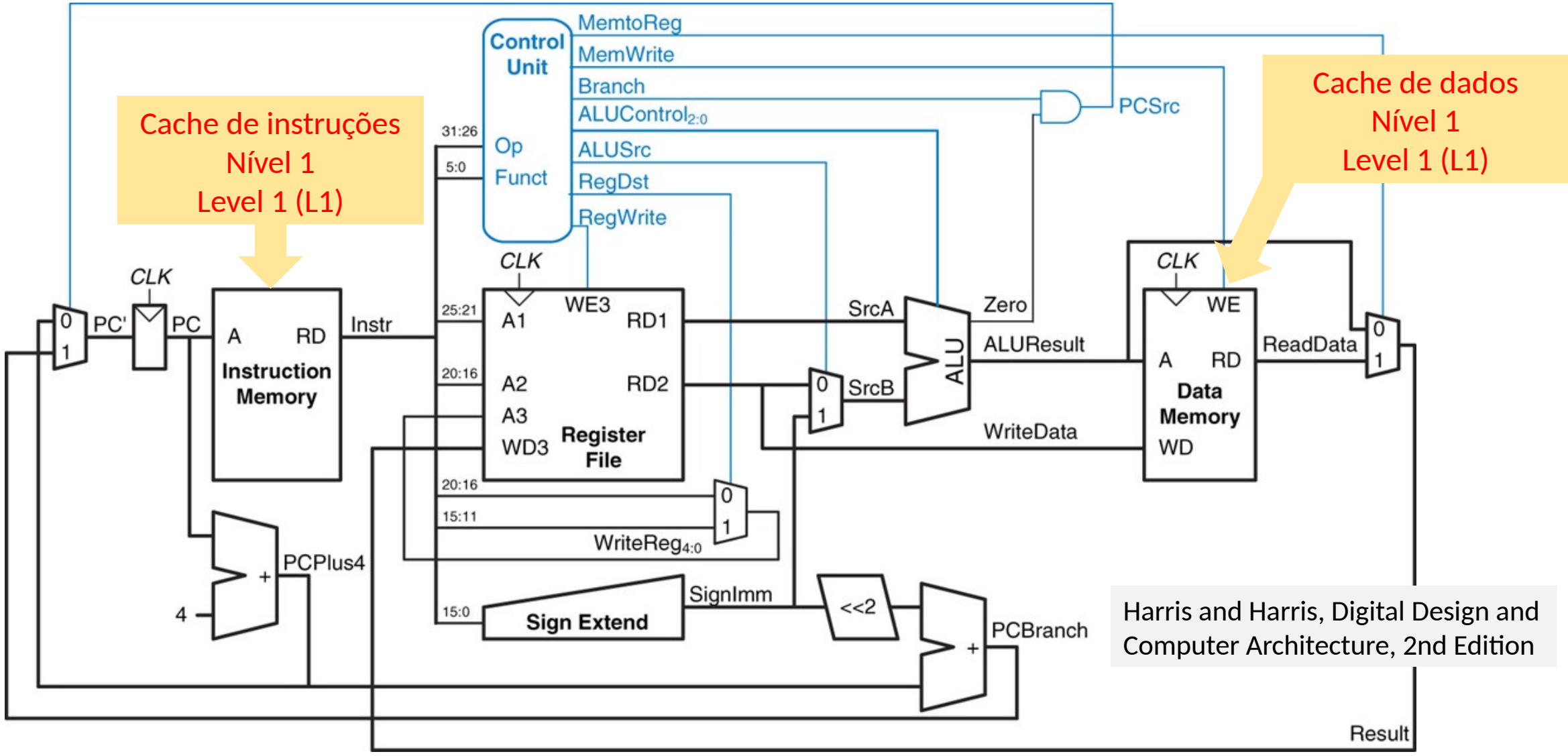
Arquitetura de Computadores III

Introdução a hierarquia de memória

Hierarquia de Memória



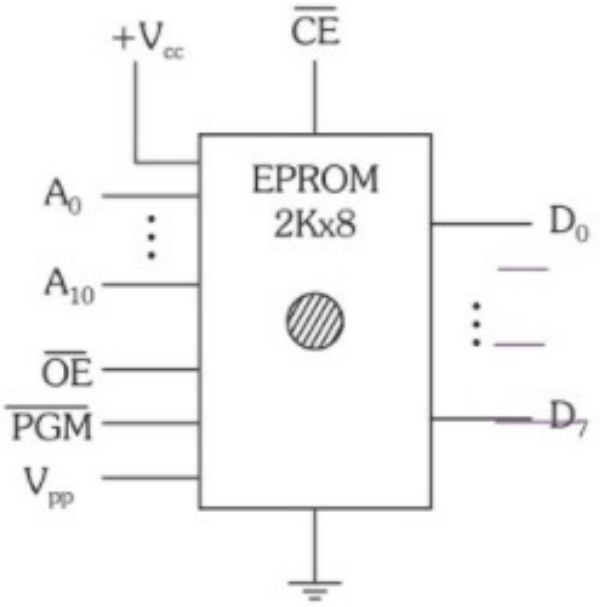
Onde aplicar os conceitos? (processador MIPS)



Harris and Harris, Digital Design and Computer Architecture, 2nd Edition

Memória

- RAM: Random-Access Memory
- ROM: Read-Only Memory
- PROM: Programmable ROM
  - EPROM: Apagável com radiação ultravioleta
  - EEPROM: Apagável por sinais elétricos.
- Registradores?



Editora Érica - Sistemas Digitais: Circuitos Combinacionais e Sequenciais - Francisco Gabriel Capuano - 1a Edição

Memória

- Célula RAM estática: A memória estática é capaz de manter os bits de dados armazenados apenas enquanto a fonte de alimentação estiver conectada ao circuito. Uma célula SRAM é equivalente ao *flip-flop*.

- Célula RAM dinâmica: A DRAM é similar a SRAM. A diferença é o projeto das células. As células dinâmicas são mais simples e necessitam de menos área no chip. Isto permite que a DRAM seja construída com densidades de armazenamento maiores, reduzindo o custo do bit. A DRAM é muito utilizada para memórias principais dos computadores. A desvantagem da DRAM é que as células são mais lentas. Os tempos de leitura e escrita são maiores. Uma célula DRAM é construída a partir de capacitores, demandando *refresh* de memória para manter o os dados armazenados.

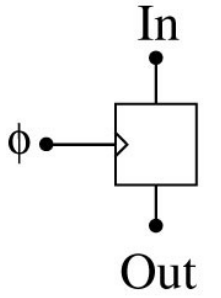
Uyemura, Sistemas Digitais, Uma abordagem Integrada, Thomson, 2002

O que é uma palavra de dados?

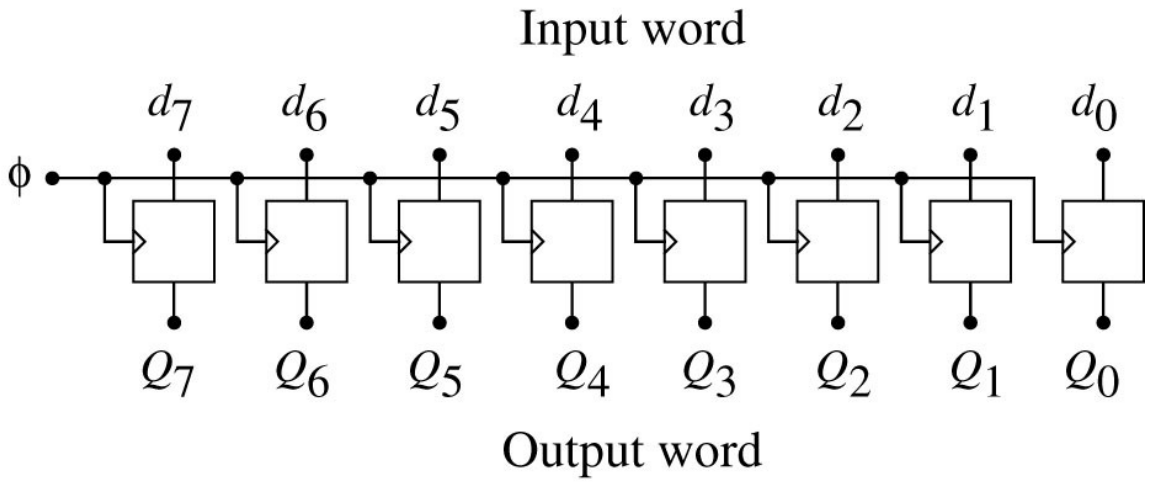
- Conjunto de bits?
- Quantos bits?
- O que significa processador de 32 bits? E de 64 bits?
- Como representar a palavra de dados?
- Como armazenar uma palavra de dados?

Registradores

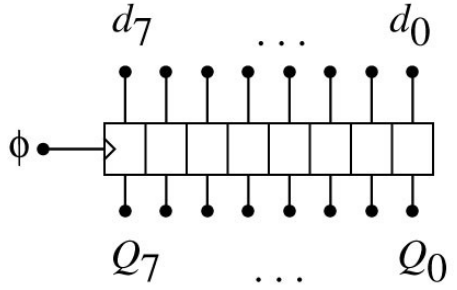
- Um registrador é um elemento lógico utilizado para armazenar uma palavra binária de n-bits.



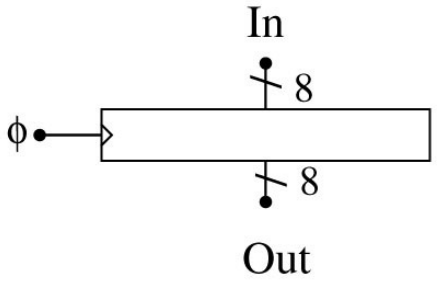
(a) Single cell



(b) 8-bit register



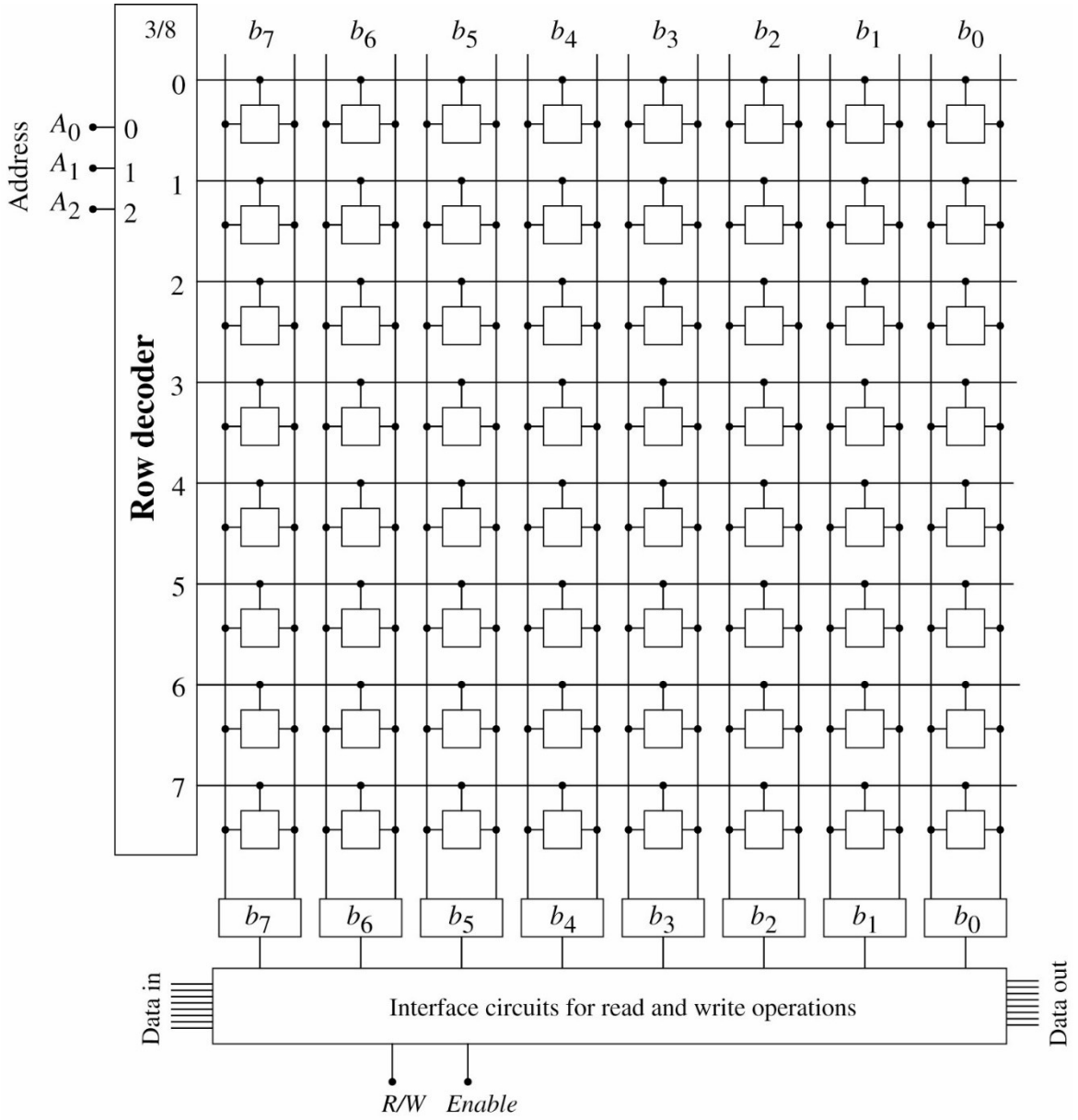
(a) Individual cells



(b) Single register

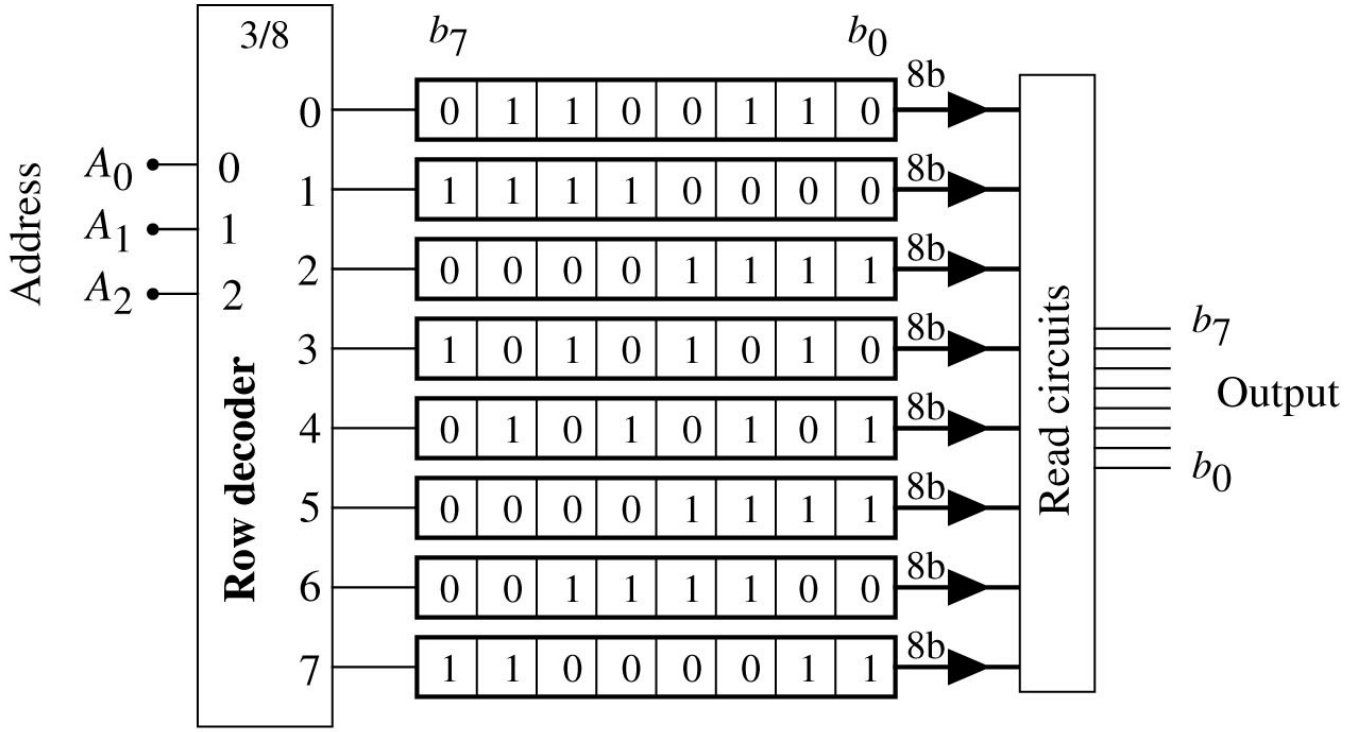
Uyemura, Sistemas Digitais, Uma abordagem Integrada, Thomson, 2002

Memória (arranjo de SRAM – matriz 8x8)



Uyemura, Sistemas Digitais, Uma abordagem Integrada, Thomson, 2002

Memória (leitura em uma matriz SRAM)



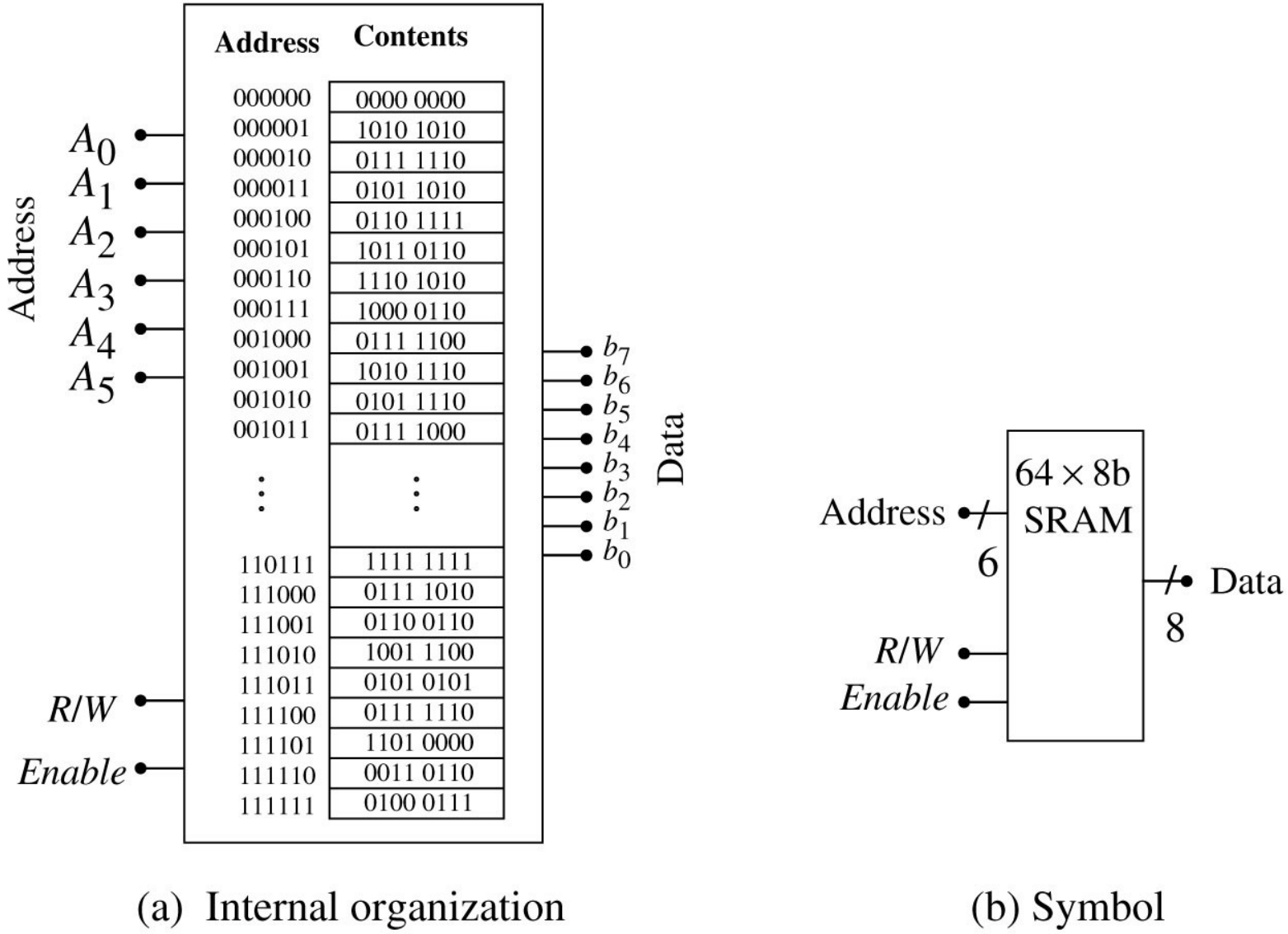
(a) Simplified network

Address $A_2\,A_1\,A_0$	Row	Output $b_7 \ldots b_0$
0 0 0	0	0110 0110
0 0 1	1	1111 0000
0 1 0	2	0000 1111
0 1 1	3	1010 1010
1 0 0	4	0101 0101
1 0 1	5	0000 1111
1 1 0	6	0011 1100
1 1 1	7	1100 0011

(b) Function table

Uyemura, Sistemas Digitais, Uma abordagem Integrada, Thomson, 2002

Memória (SRAM 64x8)



Uyemura, Sistemas Digitais, Uma abordagem Integrada, Thomson, 2002

Prefixos de unidades

Exp.	Explicit	Prefix	Exp.	Explicit	Prefix
10^{-3}	0.001	milli	10^{3}	1,000	Kilo
10^{-6}	0.000001	micro	10^{6}	1,000,000	Mega
10^{-9}	0.000000001	nano	10^{9}	1,000,000,000	Giga
10^{-12}	0.000000000001	pico	10^{12}	1,000,000,000,000	Tera
10^{-15}	0.000000000000001	femto	10^{15}	1,000,000,000,000,000	Peta
10^{-18}	0.000000000000000001	atto	10^{18}	1,000,000,000,000,000,000	Exa
10^{-21}	0.000000000000000000001	zepto	10^{21}	1,000,000,000,000,000,000,000	Zetta
10^{-24}	0.000000000000000000000001	yocto	10^{24}	1,000,000,000,000,000,000,000,000	Yotta

Como seria para a base binária?

Kilo =

Mega =

Giga =

Tera =

1ms, 1µs, 1ns

1000ps = 1ns

Prefixos de unidades

Exp.	Explicit	Prefix	Exp.	Explicit	Prefix
10^{-3}	0.001	milli	10^{3}	1,000	Kilo
10^{-6}	0.000001	micro	10^{6}	1,000,000	Mega
10^{-9}	0.000000001	nano	10^{9}	1,000,000,000	Giga
10^{-12}	0.000000000001	pico	10^{12}	1,000,000,000,000	Tera
10^{-15}	0.000000000000001	femto	10^{15}	1,000,000,000,000,000	Peta
10^{-18}	0.000000000000000001	atto	10^{18}	1,000,000,000,000,000,000	Exa
10^{-21}	0.000000000000000000001	zepto	10^{21}	1,000,000,000,000,000,000,000	Zetta
10^{-24}	0.000000000000000000000001	yocto	10^{24}	1,000,000,000,000,000,000,000,000	Yotta

Como seria para a base binária?

Kilo = 2^{10}

Mega = 2^{20}

Giga = 2^{30}

Tera = 2^{40}

1ms, 1µs, 1ns

1000ps = 1ns

Prefixos de unidades

Exp.	Explicit	Prefix	Exp.	Explicit	Prefix
10^{-3}	0.001	milli	10^{3}	1,000	Kilo
10^{-6}	0.000001	micro	10^{6}	1,000,000	Mega
10^{-9}	0.000000001	nano	10^{9}	1,000,000,000	Giga
10^{-12}	0.000000000001	pico	10^{12}	1,000,000,000,000	Tera
10^{-15}	0.000000000000001	femto	10^{15}	1,000,000,000,000,000	Peta
10^{-18}	0.000000000000000001	atto	10^{18}	1,000,000,000,000,000,000	Exa
10^{-21}	0.000000000000000000001	zepto	10^{21}	1,000,000,000,000,000,000,000	Zetta
10^{-24}	0.000000000000000000000001	yocto	10^{24}	1,000,000,000,000,000,000,000,000	Yotta

Como seria para a base binária?

Kilo = 2^{10}

Mega = 2^{20}

Giga = 2^{30}

Tera = 2^{40}

1ms, 1µs, 1ns

1000ps = 1ns

1kb, 1Mb, 1Gb

1kbps, 1kb/s

128Mb/s

Prefixos de unidades

Exp.	Explicit	Prefix	Exp.	Explicit	Prefix
10^{-3}	0.001	milli	10^{3}	1,000	Kilo
10^{-6}	0.000001	micro	10^{6}	1,000,000	Mega
10^{-9}	0.000000001	nano	10^{9}	1,000,000,000	Giga
10^{-12}	0.000000000001	pico	10^{12}	1,000,000,000,000	Tera
10^{-15}	0.000000000000001	femto	10^{15}	1,000,000,000,000,000	Peta
10^{-18}	0.000000000000000001	atto	10^{18}	1,000,000,000,000,000,000	Exa
10^{-21}	0.000000000000000000001	zepto	10^{21}	1,000,000,000,000,000,000,000	Zetta
10^{-24}	0.000000000000000000000001	yocto	10^{24}	1,000,000,000,000,000,000,000,000	Yotta

Como seria para a base binária?

Kilo = 2^{10}

Mega = 2^{20}

Giga = 2^{30}

Tera = 2^{40}

1 kb / 1kbps = ?

1 x 1024 / 1 x 1000 = 1,024s

Prefixos de unidades

V · D · E | **Múltiplos de bits** | [Esconder]

Prefixo do SI			Prefixo binário		
Nome	**Símbolo**	**Múltiplo**	**Nome**	**Símbolo**	**Múltiplo**
bit	b	10^0	bit	b	2^0
quilobit	kb	10^3	**kibibit**	Kib	2^{10}
megabit	Mb	10^6	mebibit	Mib	2^{20}
gigabit	Gb	10^9	gibibit	Gib	2^{30}
terabit	Tb	10^{12}	tebibit	Tib	2^{40}
petabit	Pb	10^{15}	pebibit	Pib	2^{50}
exabit	Eb	10^{18}	exbibit	Eib	2^{60}
zettabit	Zb	10^{21}	zebibit	Zib	2^{70}
yottabit	Yb	10^{24}	yobibit	Yib	2^{80}

Forma mais comum de contagem de múltiplos de bits ainda é pelos prefixos da SI, pois os dados são tratados pontualmente, ou seja, ou é 0 ou é 1.

https://pt.wikipedia.org/wiki/Kibibit

- Há muita discussão sobre o uso dos prefixos. O que deveria ser um padrão, ou seja, a base do prefixo é determinada pela unidade, não é levado em consideração por muitos. Por isso, há quem diga:
  - k, M, G, etc. são sempre para bases decimais
  - Ki, Mi, Gi, etc. são sempre para bases binárias
  - k, M, G, etc. possuem bases dependentes das unidades, afinal, são prefixos de unidades

Prefixos de unidades

- A discussão se k é minúsculo ou maiúsculo, se é base decimal ou binária, seria facilmente resolvida se as atenções estivessem apenas nas unidades, mantendo por padrão o k minúsculo. Exemplo:
  - 1km: unidade metro, base decimal, k = 10^3
  - 1kb: unidade bit, base binária, k = 2^{10}

- K maiúsculo na tentativa de diferenciar é desnecessário, porque a base é determinada pela unidade e não pelo K ou k. K maiúsculo é a unidade Kelvin (temperatura).

- A proposta do Ki, Mi, etc., apenas acrescenta outra forma de representar o que já possui representação. Por isso, há software, SO, artigos e livros que usam as três formas (kb, Kb e Kib), o que aumenta a atenção que deve ser dada.
  - O que deveria ser um padrão, k minúsculo e base determinada pela unidade, foi "ignorada" ao longo dos anos aumentando a confusão sobre o tema.

Palavra de dados

- 1 byte (1B) é o conjunto de 8 bits (8b).
- Uma palavra de 8 bits = 1 byte.

Tamanho da palavra	Número de valores	Abreviação para valor
8b	$2^8 = 256$	
10b	$2^{10} = 1024$	1kb
16b	$2^{16} = 65\,536$	64kb
20b	$2^{20} = 1\,048\,576$	1Mb
28b	$2^{28} = 268\,435\,456$	256Mb
30b	$2^{30} = 1\,073\,741\,820$	1Gb

Palavra de dados

- Processador com uma palavra de 16b enxerga 64k endereços.
  - Se cada endereço armazena 16 bits (2 bytes) de dados, a memória é de 128kB.

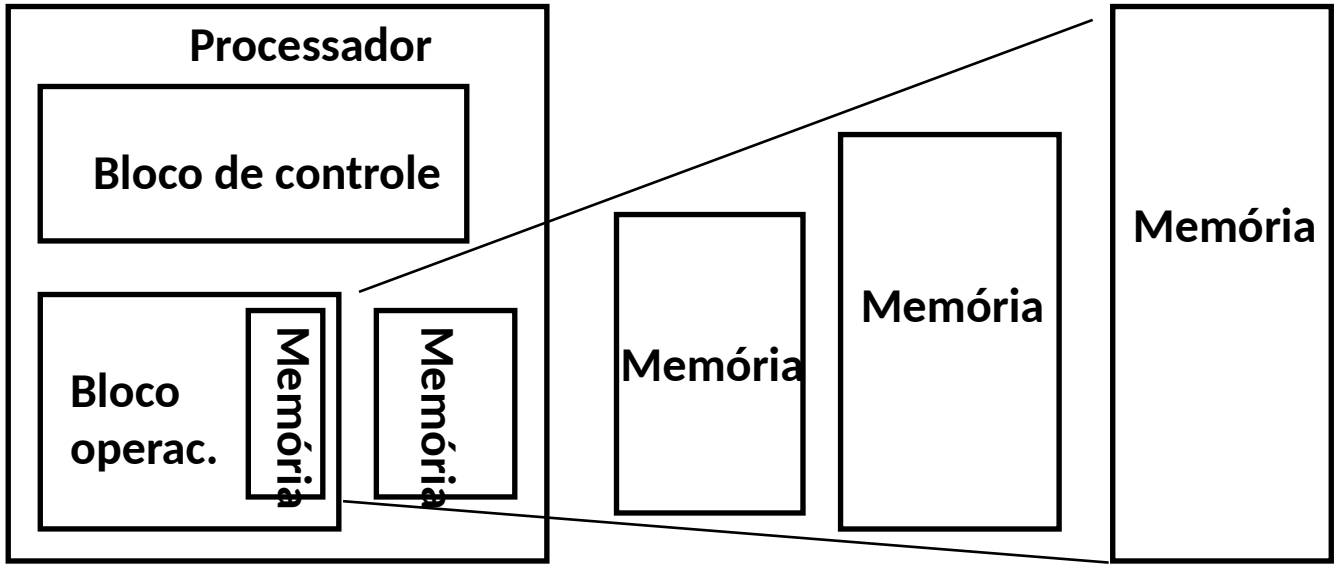
- Processador com uma palavra de 32b enxerga quantos endereços?
  - $2^{32} = 2^2 \times 2^{30} = 4 \times G = 4$Giga endereços.
  - Se cada endereço armazena 1 byte, a memória é de 4GB.

- Processador com uma palavra de 64b enxerga quantos endereços?
  - $2^{64} = 2^4 \times 2^{60} = 16 \times E = 16$Exa endereços.
  - Se cada endereço armazena 1 byte, a memória é de 16EB.

Hierarquia de Memória

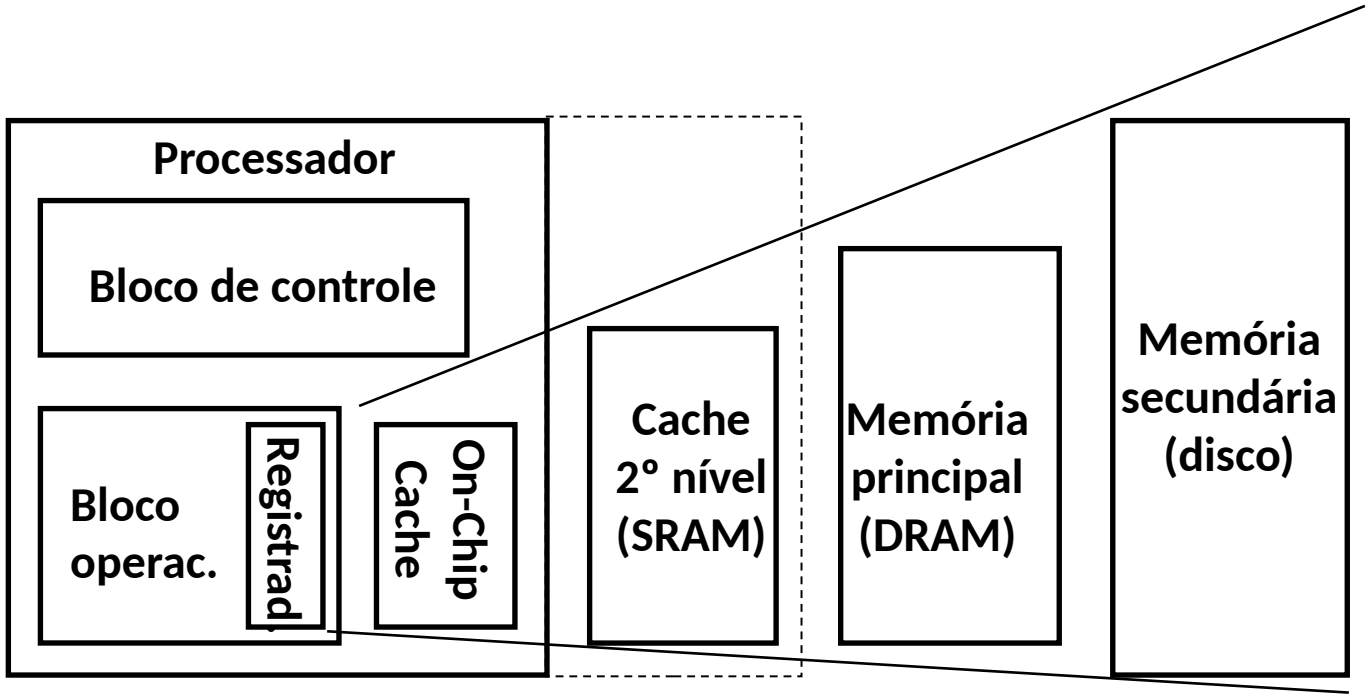
- Objetivo: oferecer ilusão de máximo tamanho de memória, com mínimo custo.
- Máxima velocidade.
- Cada nível contém cópia de parte da informação armazenada no nível superior seguinte.

Hierarquia de Memória



Velocidade:	Mais rápida	Mais lenta
Tamanho:	Menor	Maior
Custo:	Mais alto	Mais baixo

Hierarquia de Memória



"Velocidade" (ns): Tempo	0,1	1	2-5	10-20	10.000.000 (10 ms)
Tamanho (bytes):	100	16 k	512 k ...M	256 M ...Gs	G T

Hierarquia de Memória

- Como a hierarquia é gerenciada?
- Registradores <-> memória
  - pelo compilador
- cache <-> memória principal
  - pelo hardware
- memória principal <-> disco
  - pelo hardware e pelo sistema operacional (memória virtual)
  - pelo programador (arquivos)

Hierarquia de Memória

Princípio da Localidade

- Espacial: se um dado é referenciado, seus vizinhos tendem a ser referenciados logo.
- Temporal: um dado referenciado, tende a ser referenciado novamente.

Como explorar o princípio de localidade numa hierarquia de memória?

- Localidade Temporal

  => Mantenha itens de dados mais recentemente acessados nos níveis da hierarquia mais próximos do processador
- Localidade Espacial

  => Mova blocos de palavras contíguas para os níveis da hierarquia mais próximos do processador