

Hierarquia de memória e memória RAM

De uma forma genérica, costuma-se classificar as memórias utilizadas em sistemas computacionais como sendo primárias ou secundárias. As primeiras são também conhecidas como voláteis, enquanto as segundas também são conhecidas como memórias de massa. Esta classificação reflete a existência de dois grupos de mecanismos de armazenamento de dados. As memórias primárias são as utilizadas enquanto existe processamento de dados. Ou seja, enquanto o computador está ativo, os dados (e os programas) manipulados encontram-se armazenadas nestas memórias. Enquanto dados (e programas) não estão sendo utilizados, eles encontram-se armazenados em uma memória secundária. Esta hierarquia reflete a diferença tecnológica entre os mecanismos de armazenamento existentes: memórias primárias permitem acesso mais rápido, porém seu custo é mais elevado; memórias secundária permitem um valor de armazenamento por byte mais baixo, no entanto, requerem um maior tempo no acesso aos dados.

Uma representação genérica de uma hierarquia de memória em uma determinada arquitetura é dada na Figura 1. Nesta figura, existem cinco níveis de memória: registradores, memória cache interna ao processador, memória cache externa ao processador, memória principal (RAM) e disco rígido. A medida que os níveis se aproximam da CPU, diminui a disponibilidade de armazenamento do nível, enquanto aumenta o custo do byte armazenado e a velocidade de acesso aos dados.

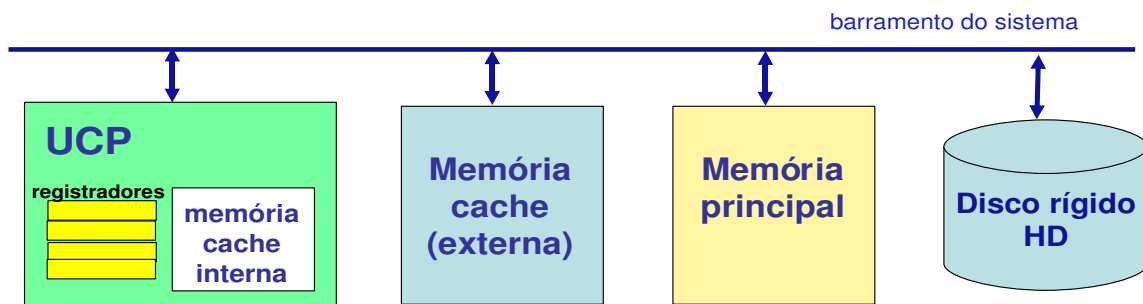


Figura 1. Hierarquia de memória em um sistema computacional.

Nesta hierarquia, a memória RAM (Random Access Memory, Memória de Acesso Randômico) merece atenção. Esta memória é composta por 2^n posições, cada posição identificada como uma *palavra* e possuindo um *endereço* próprio. A memória RAM é a representante típica de uma memória primária. Ela é utilizada para manter, durante a operação de um programa, o conjunto de dados manipulados pelo programa e o próprio código contendo as instruções deste programa. No momento em que o programa deixar de executar, todas as informações a ele pertinentes são removidas da RAM, liberando espaço para outros programas.

Uma *palavra* consiste em uma unidade de informação manipulada pela arquitetura do computador em questão, em uma determinada arquitetura, a palavra consiste em uma seqüência de m bits. Por conveniência, utilizamos $m = 8$, uma vez que um byte é a porção mínima de representação de dados. Nada impediria, no entanto, que m fosse outro valor, como 16, 32 ou 64. Na verdade, este é o caso em implementações efetivas, os primeiros computadores pessoais (PCs) utilizavam palavras de 8 bits, logo apareceram os PCs com palavras de 16 bits. Hoje já temos processadores de 64 bits disponíveis, embora os de 32 ainda dominem o mercado.

O endereço de cada palavra reflete sua posição na memória, podendo ser qualquer valor no intervalo $[0 ; 2^n - 1]$. Observe que o tamanho de memória foi indicado anteriormente como sendo dado por 2^n isto significa que o tamanho da memória é uma potência de 2. Desta forma fica evidente que o número de bits necessários para representar todas as posições de memória (ou seja, para endereça-la, é n , considerando valores inteiros e sem sinal. Esta memória é dita de acesso randômico por permitir que qualquer de suas palavras seja acessada individualmente.

O modelo de uma memória RAM é representado na Figura 2. Esta figura representa uma palavra de m bytes e tem tamanho 2^n . Esta representação contém os seguintes elementos:

- Uma seqüência de palavras endereçadas individualmente.
- Dois registradores RDM, Registrador de Dados em Memória, um para informar o dado a ser escrito na memória, outro para receber um dado lido da memória.
- Um registrador REM, Registrador de Endereço na Memória, para endereçar uma determinada palavra na memória.
- Um decodificador de endereço para selecionar uma determinada posição de memória.
- Um seletor de controle, indicando se a memória deve habilitar a palavra endereçada para receber do RDM de entrada um dado ou se deve enviar para o RDM de saída o dado presente na palavra endereçada.

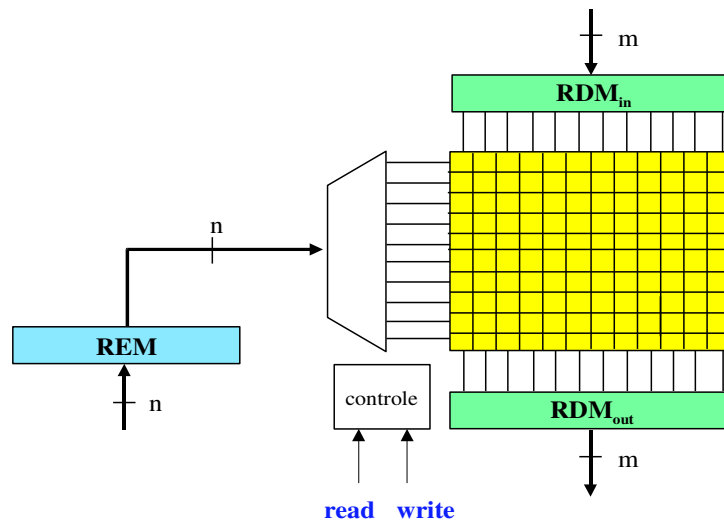


Figura 2. Modelo de uma memória RAM

A unidade de controle é responsável por comandar o funcionamento desta memória em resposta a uma instrução que está sendo executada. Por exemplo, a instrução de atribuição

```
a = 3;
```

encontrada em uma linguagem de alto nível (como C, Pascal e Perl) implica na execução dos seguintes passos:

1. O valor **3** seja carregado no RDM de entrada;
2. O endereço de memória para o qual a variável **a** está referenciando seja carregado no REM;
3. O mecanismo de controle aciona a memória para que o dado presente no RDM de entrada seja copiado para o endereço indicado no REM.

Note que o endereçamento de uma memória com 2^n palavras requer que o endereçamento seja realizado com n bits. Isto deve-se ao fato de que, havendo n bits, é possível representar 2^n valores diferentes. Considerando valores inteiros sem sinal, estes valores encontram-se no intervalo entre 0 e $2^n - 1$.

Exercícios:

1. Apresente uma definição de variável, termo empregado em linguagens de programação de alto nível.
2. O que significa uma variável considerando a memória RAM?
3. Porque os sistemas de memória em um computador são organizados de forma hierárquica? Apresente argumentos que envolvam custo (\$) e tempo de acesso.
4. O que significa memória “primária” e memória “secundária”? Cite exemplos.
5. Qual a função das memórias cache?
6. Escreva a seqüência de passos necessária para operar a memória RAM de modo a instrução **a = b;** em uma linguagem de alto nível ser bem sucedida.
7. Quantas linhas de endereçamento (ou seja, qual o comprimento em bits para representar o endereço) é necessário para uma memória que possua:
 - a) 2 palavras
 - b) 4 palavras
 - c) 16 palavras
 - d) 1024 palavras (1 K)
8. Qual o tamanho da memória que pode ser endereçado com:
 - a) 10 bits
 - b) 16 bits
 - c) 20 bits
 - d) 32 bits