Tópico 05 – SUBSISTEMA DE MEMÓRIA

Introdução: Estudar aspectos referentes ao subsistema de meória, os tipos de memória, as operações básicas, a tecnologia e assuntos relacionados a controle de erros e arquitetura de memórias. Adiantamos os assuntos relacionados a álgebra booleana a fim de descrever uma das formas de se implementar memórias ROM.

5.1 – Subsistema de Memória

Objetivo: Armazenar informações para recuperação posterior (imediata ou não).

Tipos de memória: Criados devido principalmente a diferença de velocidade com o processador, diferenças no tamanho e necessidades de armazenamento.

Representação básica: Como as casas em uma rua. Cada casa tem seu endereço (representação unívoca – 1:1)

Elemento básico de memória: O bit, que foi implementado de diversas formas como pequenas regiões magnéticas (disquetes, fitas e HDs), regiões com diferente reflexão de luz (CDs, CDs R/W e DVDs) e níveis elétricos (memórias RAM, ROM, etc.)

Célula: Como o bit apenas representa dois estados (ou dois valores distintos: 0 ou 1), criou-se a célula de memória, que um conjunto de bits que representa a informação de algum tipo. A indústria adota o tamanho de 8 bits para a célula (quase um padrão).
Ex.: Uma célula com 8 bits pode representar até 256 símbolos diferentes. Poderíamos dizer que o valor 00000000 representa o número 0 e assim por diante até o número nove(00001001). Depois poderíamos usar o valor 00001010 para representar algum caractere ou sinal de pontuação como a vírgula "," e assim por diante.

Memória Principal: Local onde os programas (ou partes dele) serão carregados para serem executados sequencialmente pelo processador (CPU-UCP). Utiliza a tecnologia DRAM (será apresentada mais adiante).

Endereço de Memória: Um endereço na memória principal é implementado através de um circuito eletrônico chamado "decodificador de endereços". Dessa forma não é necessário percorrer a lista de endereços para se encontrar o endereço desejado na memória, todos podem ser acessados com o mesmo gasto de tempo (tempo esse que é muito baixo).

Operações na Memória Principal:

Leitura (read ou retrieve) – não destrutiva
Escrita (write ou Record) – destrutiva (o conteúdo anterior se perde)

5.2 - Parâmetros utilizados na Hierarquia de Memórias:

Tempo de Acesso: Tempo entre a solicitação de um dado e a sua efetiva colocação no barramento (este parâmetro é válido para todos os tipos de memória).

Ciclo de Memória: Tempo entre um acesso e outro (este parâmetro tem sentido apenas para as memórias eletrônicas).

Ex.: Um memória DRAM (dinamic RAM) necessita dar um pulso de energia para suas células após algum acesso. Por isso, não se pode ler ou escrever até que a memória tenha executado esse pulso. Dessa forma, seu ciclo de memória é mais lento que as SRAM (static RAM) que não necessitam desse pulso.

Capacidade: Quantidade de informação a ser armazenada(este parâmetro é válido para todos os tipos de memória).

Volatilidade: Este parâmetro é válido para todos os tipos de memórias. Podemos ter dois tipos de memórias:

Volátil: Aquela que perde as informações quando deixa de receber energia (é o caso da memória principal do computador – MP – também chamada de RAM).

Não-Volátil: Não perde as informações quando fica sem energia (alimentação elétrica). Ex.: HDs, CDs, memórias ROM.

Tecnologia: Depende do emprego a ser dado à memória. Memórias eletrônicas rápidas tem pequena capacidade e possuem tecnologia diferente das memórias eletrônicas com grande capacidade. Da mesma forma a tecnologia empregada nos HDs é bem diferente da empregada nas memórias eletrônicas (este parâmetro é válido para todos os tipos de memórias).

Temporalidade: Pode ser transitória ou permanente. Registradores necessitam ficar com a informação por alguns nano segundos até ser utilizada, já na memória cachê a informação pode ficar alguns nano segundos até horas. Nos HDs a informação é permanente.

Custo: Depende principalmente da tecnologia, tempo de acesso necessário, capacidade, etc. A melhor medida é o preço relativo aos bytes armazenados (custo/byte).

Ex.: Considerando-se registradores, cachê, MP e MS, temos:
Custo/Byte: Registradores > Cache > MP > MS
Capacidade em Bytes: MS > MP > Cache > registradores

5.3 - Memória Secundária

Composta normalmente de elementos eletromecânicos (drives de CD, HD, FD, DVD, DAT, etc.). Sua capacidade sempre é muito alta se comparada com a MP e seu tempo de acesso é bem mais baixo se comparado com ela (pois a MS possui partes mecânicas e locais que necessitam ser posicionados para escrita/leitura). Possui baixo custo/byte.
Obs.: Existem memórias eletrônicas (com semicondutores) que possuem mesma tecnologia da MP mas são utilizadas como MS. Enquadram-se nesse exemplo os flash disks, os pendrives e elementos semelhantes.

5.4 - Organização da Memória Principal

A MP deve ser acessada sequencialmente pela UCP, se formos utilizar a arquitetura de Von Newmann, pois os programas também serão organizados de forma seqüencial. Porém, existe uma diferença crucial entre a estrutura da MP e da UCP (CPU): Enquanto a MP (e a cachê) é organizada em células, a UCP organiza as informações em PALAVRAS. Essa diferença se dá pelo fato de termos empresas diferentes produzindo microprocessadores e memórias.

Normalmente a palavra tem tamanho muito maior que a célula e a tecnologia deve ser capaz de acomodar as diferenças entre os sistemas. De forma simplificada a **UCP** envia/recebe uma palavra e a **MP**/cachê deve acomodar essa palavra em um conjunto de células.

Ex.: Palavra da UCP (32bits) deve ser acomodada em 4 células (8bits cada) na MP/cachê.

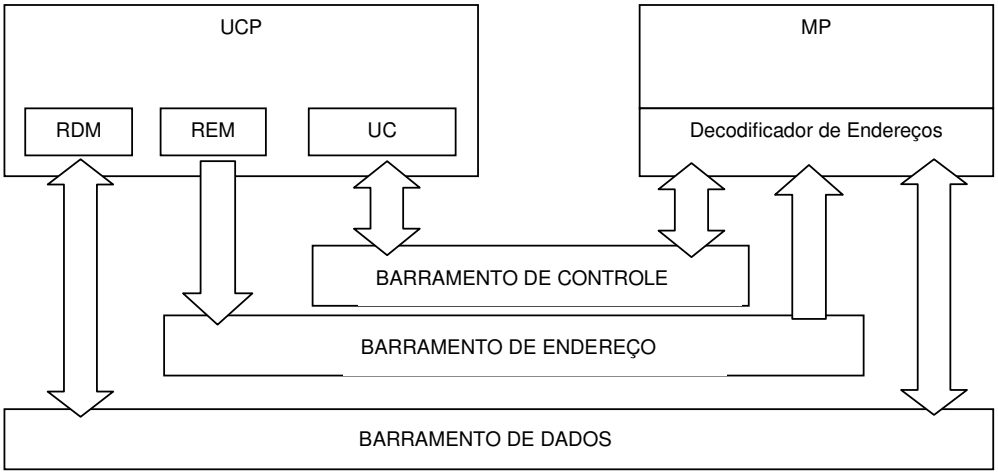
Unidade de Transferência: Tomemos como exemplo uma UCP que possui palavra de 32 bits e barramento de dados de 64 bits. Numa operação de escrita/leitura serão movimentadas 2 palavras de/para a UCP, resultando em 8 células lidas/escritas. Dessa forma a unidade de transferência irá depender da largura do barramento de dados/instruções.

Tamanho em bits: NxM onde:
N = qtd de células
M = qtd de bits por célula
Ex.: Uma memória endereço de 8bits → N = 2^8 = 256 posições, se M = 8 => T = 256x8 = 2048 bits no total

0	00	00000000	Célula 0
:::	:::	:::	:::
255	FF	11111111	Célula 255

Note que as células são numeradas de 0 a $2^N - 1$ (0 a 255)

Esquema Básico da Comunicação UCP – MP:



Operação de Leitura: Formada por micro-operações

1)(REM) ← (PC) [contador do programa coloca endereço no REM]
1a)Endereço é colocado no barramento de endereços
2)Sinal de leitura é colocado no barramento de controle
2a)Decodificação do endereço e localização da célula
3)(RDM) ← MP(REM) pelo barramento de dados [MP envia dado]

Operação de Escrita: Formada por micro-operações
1)(REM) ← (PC) [contador do programa coloca endereço no REM]
1a)Endereço é colocado no barramento de endereços
2)(RDM) ← (outro registrador)
3)Sinal de escrita é colocado no barramento de controle
4)(MP(REM)) ← (RDM)

Obs.: Os comandos foram escritos em **RTL**(LTR - Linguagem de Transferência entre Registros).

5.5 - Cálculos da Capacidade de Memórias

Para executar os cálculos de capacidade de memórias vale lembrar os múltiplos K(2^{10}), M(2^{20}), G(2^{30}), T(2^{40}) e P(2^{50}) para a base 2. Dados:
N = Qtd de células
E = qtd de bits de cada endereço => $N = 2^E$
$T = 2^E \times M$

Ex.1: 512 células necessitam de quantos bits para serem endereçadas?
$N = 512 = 2^E => 2^9 = 2^E => E = 9$ bits p/ endereçar 512 posições

Ex.2: Se cada célula tiver 8 bits, qual será a capacidade dessa memória?
$T = 2^9 x 8 = 2^9 x 2^3 = 2^{12} = 2^2 x 2^{10} = 4$Kbits (512bytes)

Ex.3: Um tipo de RAM com células de 16bits e 2K células, qual é a capacidade total
$T = 2x2^{10}x16 = 2x2^{10}x2^4 = 2x2^4x2^{10} = 32x2^{10} = 32$Kbits (4Kbytes)

5.6 - Tipos e Nomenclatura da MP

Com o avanço da tecnologia mais e mais tipos de memórias a semicondutor foram sendo criadas. A seguir descreveremos de forma simples alguns desses tipos.

RAM – Random Access Memory – Memória de Acesso Aleatório: Este termo é comumente empregado para memórias a semicondutor voláteis (normalmente as DRAM usadas como MP), mas todas as memórias a semicondutor são de acesso aleatório. Isso significa que qualquer endereço pode ser acessado com o mesmo gasto de tempo (ciclo de memória) como dissemos anteriormente. O acesso não depende da posição do endereço na memória.

> **DRAM** – Dynamic RAM – Necessitam receber um pulso de energia (refresh) de tempos em tempos para não perder os dados armazenados e também quando um dado é escrito/lido. Este tipo de memória é usado na MP.
>
> **SRAM** – Static RAM – Sua tecnologia permite serem acessadas sem a necessidade do refresh. Dessa forma são mais rápidas que as DRAM e por isso mais caras. Esse tipo de memória é usado na memória cachê.

ROM – Read Only Memory – Memórias de Leitura Apenas (memórias de apenas leitura) – Também é uma memória de acesso aleatório com as RAM, mas sua tecnologia de fabricação é bem mais simples, pois seu emprego é específico para a gravação de microprogramas como o Sistema Básico de Entrada e Saída (BIOS). O BIOS é usado no IPL (initial program load) ou ”boot” do computador. Outros dispositivos periféricos utilizam memórias ROM para guardar seus microprogramas (também conhecidos como firmware). Até mesmo os pentes de memória RAM possuem pequenas memórias ROM para controle de atividades.

PROM – Programmable ROM – ROM programável – Este tipo de ROM sai da fábrica sem nenhum programa. Outras indústrias a compram para inserir seus próprios firmwares. Uma vez feita a gravação não se pode mais modificar o conteúdo.

EPROM – Erasable PROM – PROM apagável - Este tipo de chip possui uma janela (abertura coberta por um plástico transparente) por onde se pode aplicar uma luz ultra-violeta de alta intensidade que irá apagar todo o programa, possibilitando nova gravação.

EEPROM – Electrically EPROM – EPROM apagável eletricamente – Este tipo de chip possibilita alterações no código byte a byte através da aplicação de sinais elétricos em determinados pontos do mesmo.

FLASH ROM – Funciona como uma EEPROM (ou E2PROM) mas não apaga byte a byte. Sua vantagem é possibilitar um apagamento muito rápido, dessa forma ela se comporta como um a RAM.

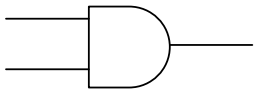
5.7 - Tecnologia de construção das ROMs

Inicialmente construiu-se ROMs à partir de estruturas miniaturizadas com muitos resistores ou diodos. Cada bit a ser gravado deveria ser “queimado” na ROM. Se o bit era formado por um diodo, aplicava-se uma alta tensão reversa para fazer com que o mesmo se queimasse, fazendo daquela posição um circuito aberto (bit 0). Com a tecnologia de resistores a situação era semelhante. Após a programação, não seria possível regravar a ROM.
ROMs com portas lógicas: Outra forma de criar ROMs é utilizar a tecnologia de portas lógicas. Os circuitos lógicos seriam criados na ROM e uma vez montados não seria possível desfazê-los.

Portas Lógicas e Álgebra Booleana: A fim de entender este tipo de tecnologia empregada na criação de ROMs, outros circuitos eletrônicos e também na programação de computadores, iremos explicar de forma simples as portas lógicas e a álgebra booleana (assunto relacionado a tipos de dados).

Porta E (AND)

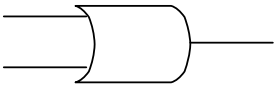
Uma porta E define uma comparação de sinais onde a saída só será verdadeira se todas as entradas forem verdadeiras, independentemente do número de entradas. **Verdadeiro** é representado por **1** e **Falso** é representado por **0**.



Ex.: Comparando-se bit a bit **00110** e **10010**, o circuito terá como resposta **00010**:
00**1**1**0**
<u>**1**00**1**0</u>
000**1**0 Pois apenas os bits correspondentes à segunda casa são verdadeiros (1 em ambos).

Porta OU(OR)

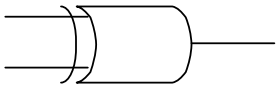
Uma porta OU define uma comparação de sinais onde a saída será verdadeira desde que pelo menos uma entrada seja verdadeira, independentemente do número de entradas. **Verdadeiro** é representado por **1** e **Falso** é representado por **0**.



Ex.: Comparando-se bit a bit **00110** e **10010**, o circuito terá como resposta **10110**
00**11**0
<u>**1**00**1**0</u>
10**11**0 Pois pelo menos um dos bits correspondentes às casas 2,3 e 5 são verdadeiros.

Porta OU Exclusivo (XOR - XOU)

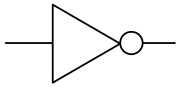
Uma porta XOU define uma comparação de sinais onde a saída só será verdadeira se uma das entradas for verdadeira, independentemente do número de entradas. **Verdadeiro** é representado por **1** e **Falso** é representado por **0**.



Ex.: Comparando-se bit a bit 00110 e 10010, o circuito terá como resposta 10110
00**11**0
<u>**1**00**1**0</u>
10**10**0 Pois apenas nos bits correspondentes às casas 3 e 5 temos 1 bit 1.

Porta NÃO(NOT)

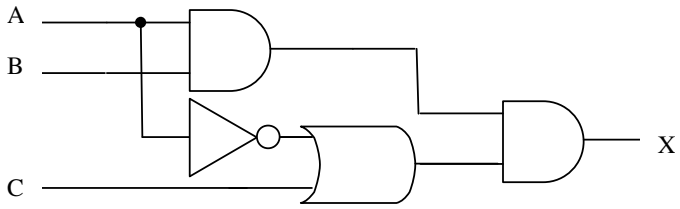
Uma porta NÃO inverte a lógica dos bits, onde era verdadeiro (1) se torna falso (0) e vice-versa, independentemente do número de entradas.



Ex.: Invertendo-se bit a bit **00110**, o circuito terá como resposta **11001**

Álgebra Booleana: Quando se necessita criar circuitos lógicos com muitas entradas e diversas operações nessas entradas, deve-se utilizar a álgebra booleana a fim de organizar o raciocínio lógico, tanto na montagem do circuito, quanto nas funções lógicas de um programa. Basicamente a álgebra boolena obedece as convenções da álgebra tradicional, porém aplicada às operações lógicas implementadas pelas portas lógicas.

Ex.: Deseja-se criar um circuito lógico com 3 entradas: A, B e C. Este circuito está simbolizado pelo diagrama a seguir. X será a saída desse circuito.



A expressão booleana que substitui o circuito é: X = (A **E** B) **E** ($\overline{A}$ **OU** C). Veja o inverso da entrada A é simbolizada com $\overline{A}$. Podemos reescrever a expressão da seguinte forma:

X = (A• B) •$\overline{(A + C)}$ = (A.B).($\overline{A}$+C) – dessa forma fica mais parecido com uma expressão algébrica.

Agora, se A = 0010, B = 1110, C = 1010, teremos

A	$\overline{A}$	B	C	A.B	$\overline{A+C}$	X
0	1	1	1	0	1	0
0	1	1	0	0	1	0
1	0	1	1	1	1	1
0	1	0	0	0	1	0

Portanto, **X = 0010**.

5.8 - Checagem de Erros nas memórias eletrônicas

Devido aos fatores elétricos inerentes a construção dos elementos da memória e das linhas usadas nos barramentos (atenuação de sinal, etc.), o processo de escrita/leitura pode falhar. Dessa forma, faz-se necessário criar algum mecanismo de checagem de erros. Uma forma simples (de certo modo) é a inserção de bits de checagem de erros em cada célula. Ao se armazenar um valor gera-se o bit (ou bits) de checagem de erros através de uma operação matemática com os bits do valor armazenado. Ao se tentar recuperar o valor armazenado, faz-se novamente a operação matemática de checagem de erro e compara-se os dois valores gerados, se forem iguais a recuperação foi correta.

5.9 - Memória Cache

Este tipo de memória foi criada principalmente para compatibilizar a velocidade da UCP e da MP. Originalmente a UCP tem que aguardar alguns ciclos de máquina (clocks) para possibilitar que a MP lei/escreva o dado solicitado. Este tempo de espera é chamado de estado de espera (wait state).
A memória cachê é construída com SRAMs e se vale do princípio da localidade (tanto espacial quanto temporal).

Localidade Espacial
Se a UCP acessou uma palavra da memória em um determinada parte (bloco) do programa, existe grande probabilidade de a próxima palavra a ser acessada estar nesse mesmo bloco.

Localidade temporal
Se uma palavra foi acessada pela UCP, existe grande probabilidade dessa palavra ser acessada novamente em um breve espaço de tempo.

Operação de Acesso à cachê

1) A UCP acessa o início do programa que está na MP
2) Esse bloco será transferido para a cachê (cachê p/ dados e cache p/ instruções)
3) Os demais acessos serão feitos a cachê. Se a palavra a ser acessada for encontrada na cachê teremos um hit
4) Caso a palavra não esteja na cachê (perdida - miss), o sistema interrompe a execução do programa e carrega o bloco com a palavra necessária.

Ex.: Um sistema executou 100 tentativas de acesso a cachê. Dessas, houve 98 acertos (hits) e apenas duas palavras perdidas (miss) que resultaram am carga da MP.

98/100 = 98% (hits)
02/100 = 02% (cachê miss)

Se o tempo de acesso a cachê é de 1ns e o tempo de acesso a MP é de 100ns, nessas 100 tentativas teremos:

98x1ns + 2x100ns = 298ns no total ou 298/100 = 2,98ns (tempo médio p/ cada acesso)