Memórias

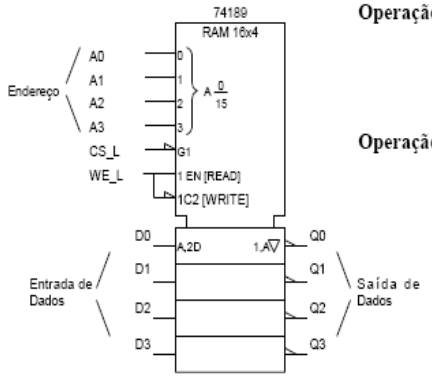
Na seqüência do estudo de **FFs**, como elementos básicos de memória de 1 bit, e de **registros** como elementos capazes de armazenar 1 palavra de N bits, surge, naturalmente, a necessidade de elementos que permitam o armazenamento conjunto de K palavras de N bits. Esses elementos designam-se por **memórias**.

Tipos de memórias

- MEMÓRIAS INTEGRADAS
  - RAM
  - ROM
  - PROM
  - EPROM
- LÓGICA PROGRAMÁVEL
  - PLAs
  - PALs
  - FPGAs

RAM -MODELO SIMPLIFICADO

RAM – "Random Access Memory"



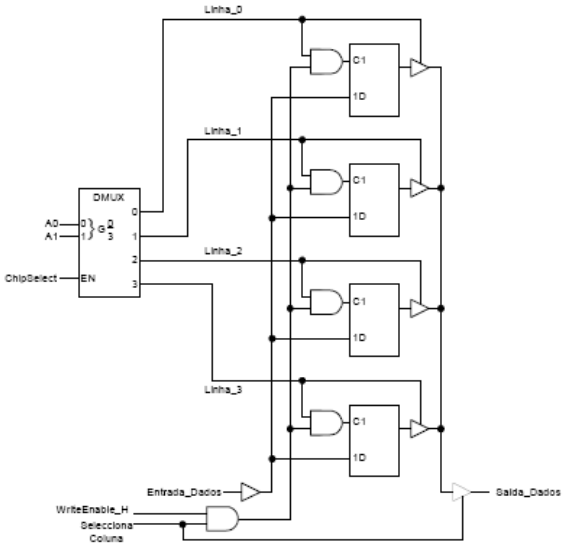
Operação de Escrita:

(1) Colocar **endereço** nas linhas de endereço;
(2) Colocar dados nas linhas de entrada de dados (Ds);
(3) Activar CS_L e WE_L (L - Escrita).

Operação de Leitura:

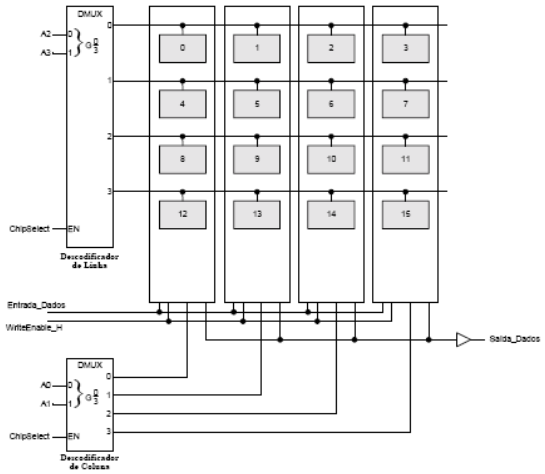
(1) Colocar **endereço** nas linhas de endereço;
(2) Activar CS_L e Desactivar WE_L (H – Leitura) ;
(3) Ler dados pretendidos nas linhas de saída (Os).

RAM - MODELO LÓGICO DA ESTRUTURA BASE



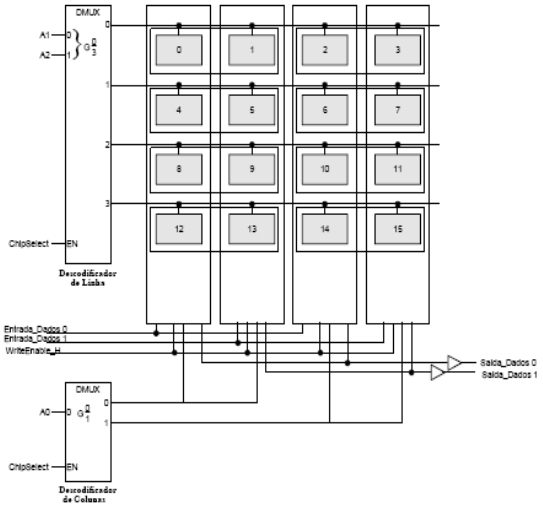
RAM - ESTRUTURA TIPO

EXEMPLO:
RAM 16x1, organizada segundo uma matriz de 4x4 bits.



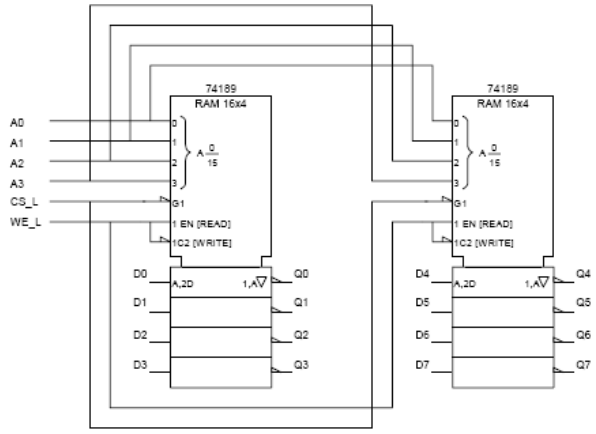
RAM - ESTRUTURA TIPO

EXEMPLO:
Uma RAM 8x2 é organizada segundo a mesma matriz de 4x4 bits, que é agora, de facto, uma matriz de 4x2 palavras de 2 bits cada.



RAM - ASSOCIAÇÃO

Aumento da Dimensão das Palavras (de 4 para 8 bits):



Tipos de memórias

- ROM

Memórias ROM ("Read-Only Memory")

As memórias ROM, também designadas por memórias mortas, são constituídas por uma matriz de dispositivos com capacidade para armazenar um bit de informação. Esta matriz é organizada por células/palavras constituídas por um número determinado de bits. São memórias não voláteis, isto é, não perdem a informação que armazenam depois de retirada a alimentação. Depois de gravadas/programadas (armazenamento da informação digital binária), estas memórias apenas permitem operações de leitura, resultando desta característica a designação ROM ("Read-Only Memory").

Tipos de memórias ROM

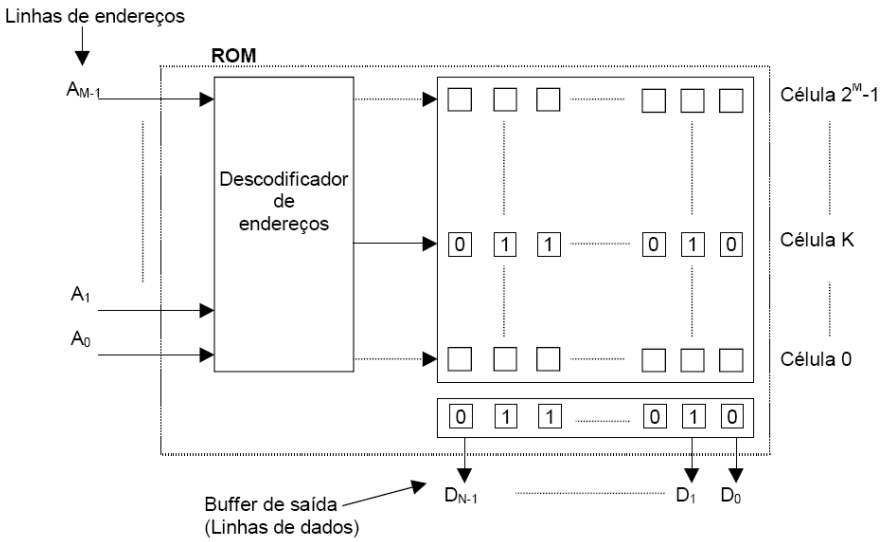
♦ **ROM** – o conteúdo das células é gravado/programado durante o processo de fabricação. Não podem ser apagadas, não sendo possível a sua reutilização.

♦ **PROM** ("Programmable ROM") – o conteúdo das células é gravado/programado pelo utilizador recorrendo a um programador (Programador de PROM's). Apenas podem ser gravadas uma única vez, não podendo ser reprogramadas tendo em vista a sua reutilização.

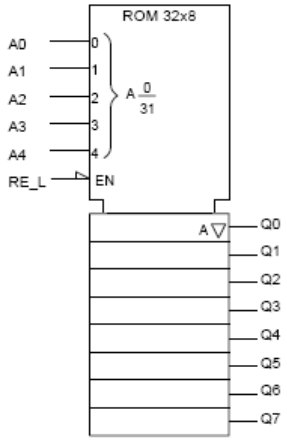
♦ **EPROM** ("Erasable Programmable ROM") – o conteúdo das células é gravado/programado pelo utilizador recorrendo a um programador (Programador de EPROM's). Podem ser apagadas através da exposição a raios ultra-violetas durante um período de tempo (5 a 20 minutos). Neste processo todos os bits da memória são colocados a 1, sendo possível a sua reutilização.

♦ **EEPROM** ("Electrically Erasable Programmable ROM") – idêntico à EPROM mas com a possibilidade de ser reprogramada através de sinais eléctricos no próprio sistema de destino.

Estrutura interna



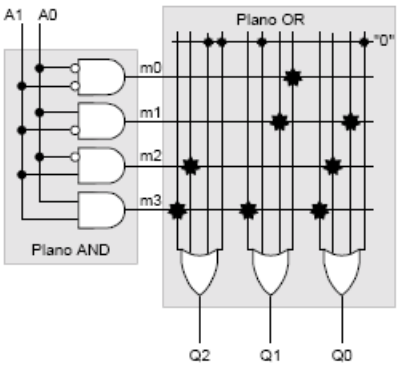
ROM - SIMBOLOGIA



Exemplo

EXEMPLO: ROM 4x3 que armazena os 4 primeiros números primos:

A_1	A_0	Q_2	Q_1	Q_0	
0	0	0	1	0	**2**
0	1	0	1	1	**3**
1	0	1	0	1	**5**
1	1	1	1	1	**7**



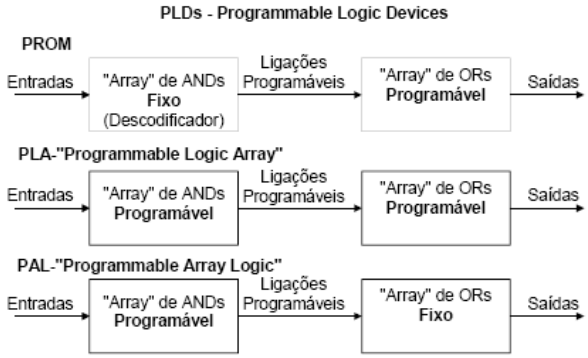
Aplicação

Utilização de memórias ROM na implementação de circuitos

As memórias ROM constituem o primeiro exemplo entre vários tipos de dispositivos de lógica programável **(PLD-Programmable Logic Devices).**

Lógica programável

- **Nesta seção apresenta-se algumas** estruturas**, derivadas da estrutura básica de uma ROM, cujo objetivo está centrado na sua programação** para implementação de funções combinatórias **e não na sua utilização como memória.**



Implementação de circuitos combinacionais

Qualquer função combinacional é diretamente realizável desde que se disponha de um circuito integrado ROM cujo número de linhas de endereços é igual ou superior ao número de variáveis da função. A utilização de ROM's na implementação de circuitos combinacionais é um processo bastante simples. De fato, uma ROM com M linhas de endereços e N linhas de dados corresponde à implementação direta de uma tabela - verdade com M variáveis de entrada e N funções de saída. Neste tipo de solução não é necessário efetuar qualquer processo de simplificação das funções de saída.

Exemplo:

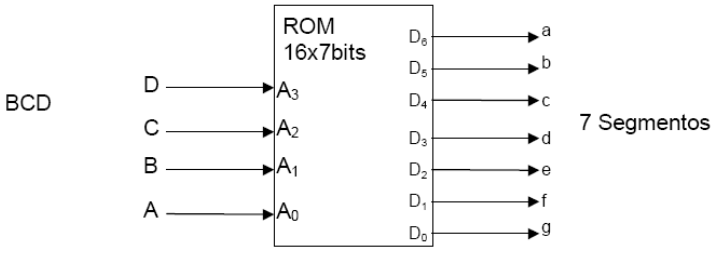
Circuito conversor BCD-7 segmentos
Variáveis de entrada: DCBA (ativas a 1)
Variáveis de saída: segmentos a, b, c, d, e, f, g (ativos a 0)

Para a implementação deste circuito combinacional é necessária uma memória ROM com 4 linhas de endereços (24=16 células de memória) e 7 linhas de dados (células de memória constituídas por 7 bits), a que corresponde uma capacidade total de 16x7 bits.

Tabela de verdade /Conteúdo das células de memória

BCD					7 Segmentos						
Nº	D	C	B	A	a	b	c	d	e	f	g
Endereço					Conteúdo						
	A_3	A_2	A_1	A_0	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	1	1	0	0	1	1	1	1
2	0	0	1	0	0	0	1	0	0	1	0
3	0	0	1	1	0	0	0	0	1	1	0
4	0	1	0	0	1	0	0	1	1	0	0
5	0	1	0	1	0	1	0	0	1	0	0
6	0	1	1	0	1	1	0	0	0	0	0
7	0	1	1	1	0	0	0	1	1	1	1
8	1	0	0	0	0	0	0	0	0	0	0
9	1	0	0	1	0	0	0	1	1	0	0
	1	0	1	0	x	x	x	x	x	x	x
	1	0	1	1	x	x	x	x	x	x	x
	1	1	0	0	x	x	x	x	x	x	x
	1	1	0	1	x	x	x	x	x	x	x
	1	1	1	0	x	x	x	x	x	x	x
	1	1	1	1	x	x	x	x	x	x	x

Diagrama lógico

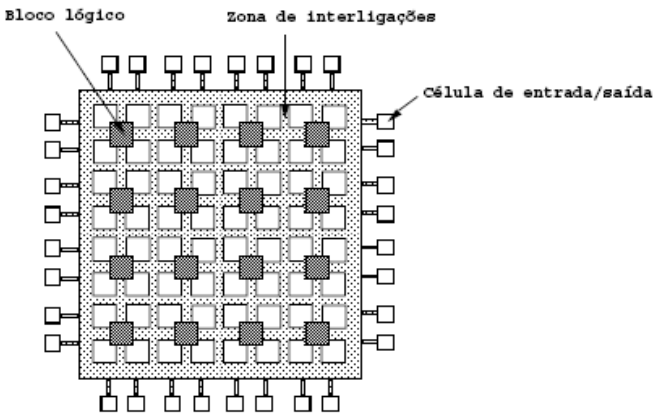


FPGAs – "FIELD PROGRAMMABLE GATE-ARRAYS"

As FPGAs são circuitos programáveis que permitem realizar circuitos de complexidade equivalente até algumas centenas de milhar de portas lógicas.

O tipo de programação usado nas FPGAs mais populares baseia-se na utilização de memórias RAM distribuídas para realizar as funções lógicas e controlar as interligações do circuito.

FPGAs – "FIELD PROGRAMMABLE GATE-ARRAYS"



Exercício:

Uma aplicação importante para ROMs é a geração de sinais de controle e temporização.

A figura 1 mostra uma ROM de 16x4 com suas entradas de endereços acionadas por um contador de módulo 16, de modo que os endereços da ROM são incrementados a cada pulso de entrada. Suponha que a ROM é programada como na tabela 1, esboce, na figura 2, as formas de onda de cada uma das saídas da ROM conforme os pulsos de clock vão sendo aplicados.

Preencha na Tabela 2 os dados a serem gravados em outra pastilha igual para obtermos as formas de onda mostradas na figura 3. Ignore os tempos de atraso da ROM. Considere que o contador começa em 0000.

Exercício:

Figura 1:

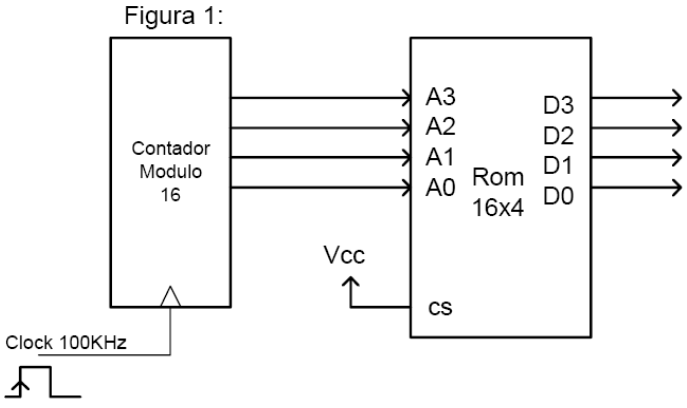
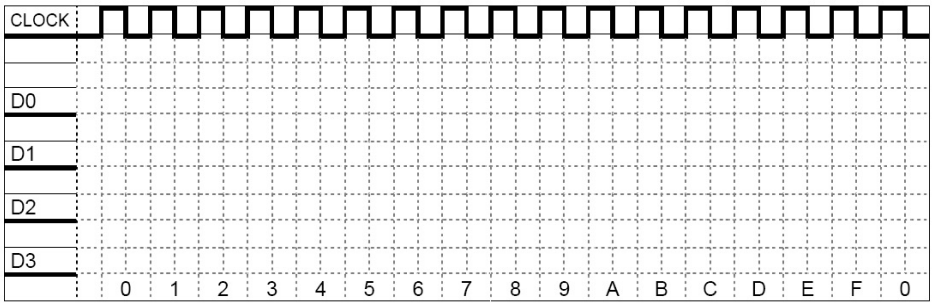


Tabela 1	
End	Dado
0	0
1	6
2	0
3	0
4	0
5	8
6	0
7	8
8	0
9	9
A	1
B	D
C	3
D	9
E	1
F	1

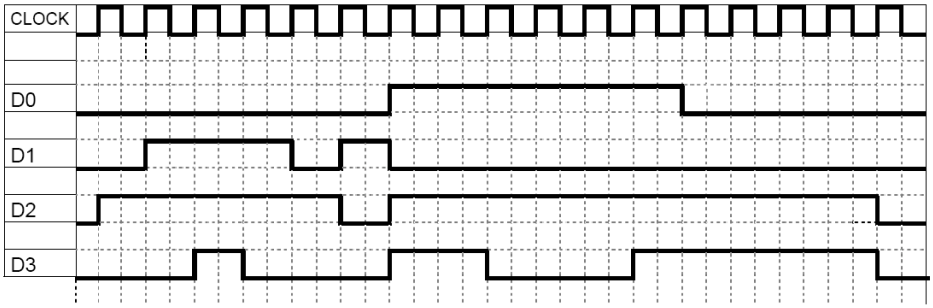
Exercício:

Figura 2:



Exercício:

Figura 3:



Exercício:

Tabela 2 Preencher essa tabela com os valores em Hexadecimal dos dados a serem gravados na memória para as formas de onda da figura 3.

HEX																
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F