Sistemas Digitais I
LESI :: 2º ano

CPLDs e Memórias

António Joaquim Esteves
João Miguel Fernandes

www.di.uminho.pt/~aje

Bibliografia: anexo C do guia TP; secções 10.1 a 10.4, DDPP, Wakerly

DEP. DE INFORMÁTICA
ESCOLA DE ENGENHARIA
UNIVERSIDADE DO MINHO

8. CPLDs e Memórias

- Sumário -

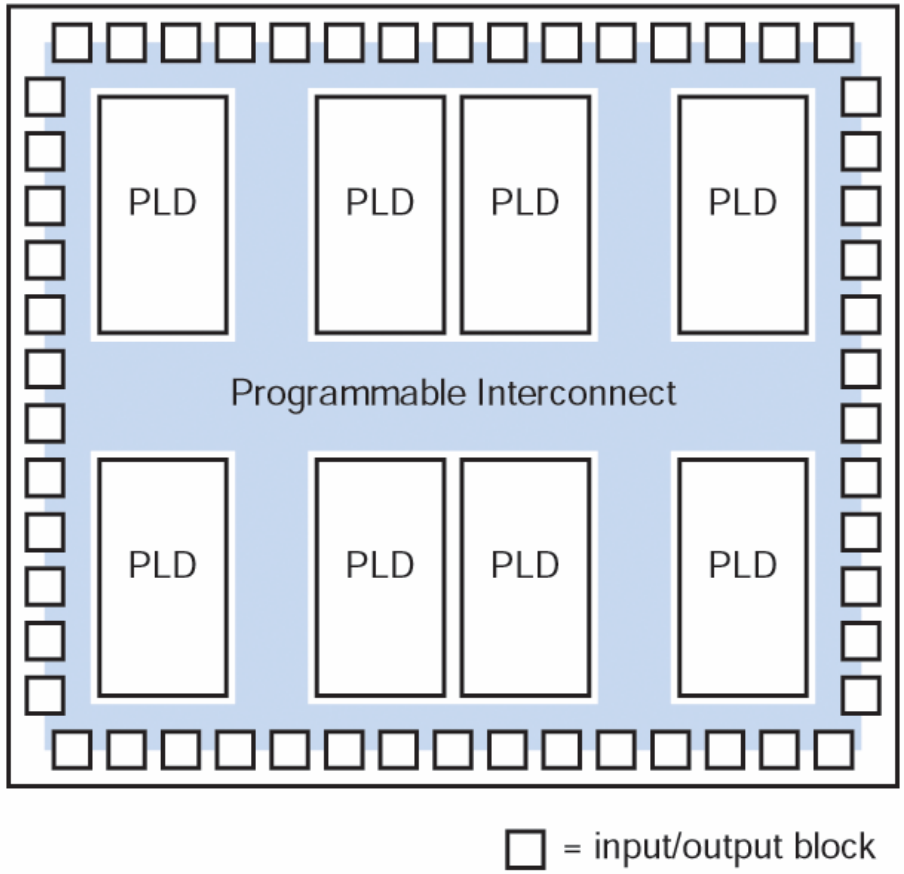
- PLDs complexas (CPLDs)
- Memórias: ROMs e RAMs

8. CPLDs e Memórias

- CPLDs (1) : introdução -

- À medida que a tecnologia de fabrico de CIs avançou, surgiu o interesse natural em produzir PLDs cada vez maiores para tirar partido duma densidade de transistores (dentro do *chip*) cada vez maior.

- Uma CPLD é um dispositivo que incorpora num único *chip* uma colecção de PLDs interligadas por uma estrutura programável.
- Esta estrutura permite que as PLDs sejam interligadas do mesmo modo que o seriam fora do *chip*.
- A família de CPLDs 9500 da Xilinx vai ser usada como exemplo para a arquitectura duma CPLD.



8. CPLDs e Memórias

- CPLDs (2) : como expandir a arquitectura das PLDs simples? -

- Hipótese de expansão: aumentar o número de entradas e saídas duma PLD convencional, ou seja, partindo das PALs 16V8, 20V8, 22V10 poderia chegar-se a PALs mais densas, tais como 32V16 ou 128V64.
- Problemas desta alternativa:
  - Aumentar *n* vezes o número de entradas e saídas exige n^2 vezes mais espaço no *chip* → logo é uma solução dispendiosa

    [*array* de ANDs tamanho (a*b) → *array* de ANDs tamanho (n*a)*(n*b)=n^2 *(a*b)]
  - Quando se aumenta o número de entradas, a lógica combinacional fica cada vez mais lenta dado que o número de entradas do *array* de ANDs aumenta.
- Solução: várias PLDs interligadas por 1 estrutura programável relativamente reduzida.
  - Esta arquitectura é menos genérica do que uma PLD de grande dimensão, mas a utilização duma ferramente ***fitter*** liberta o projectista da tarefa de atribuir a cada bloco tipo PLD (da CPLD) uma parte do sistema a implementar.

8. CPLDs e Memórias

- CPLDs (3) : famílias de CPLDs -

- Tendo por base o mesmo bloco PLD (designado por **Bloco Funcional** na Xilinx) pode construir-se variantes da mesma família de CPLD diferindo no:
  - Número de blocos PLD
  - Número de pinos de entrada/saída (I/O)
- Muitas CPLDs possuem menos pinos de I/O do que células base (**macrocélulas**):
  - Algumas macrocélulas fornecem lógica interna mas não ligam as saídas ao exterior do *chip*.
  - É possível obter encapsulamentos com um número de pinos diferente, mas com a mesma lógica interna.
  - É possível obter CPLDs com o mesmo encapsulamento, mas com lógicas internas diferentes → Tabela do próximo slide.

8. CPLDs e Memórias

- CPLDs (4) : CPLDs do fabricante Xilinx -

	Part Number					
	XC9536	**XC9572**	**XC95108**	**XC95144**	**XC95216**	**XC95288**
FBs / macrocells	2 / 36	4 / 72	6 / 108	8 / 144	12 / 216	16 / 288
Package	**Device I/O Pins**					
44-pin VQFP	34					
44-pin PLCC	34	34				
48-pin CSP	34					
84-pin PLCC		69	69			
100-pin TQFP		72	81	81		
100-pin PQFP		72	81	81		
160-pin PQFP			108	133	133	
208-pin HQFP					166	168
352-pin BGA					166	192

a mesma lógica interna e diferentes encapsulamentos

o mesmo encapsulamento e lógicas internas diferentes

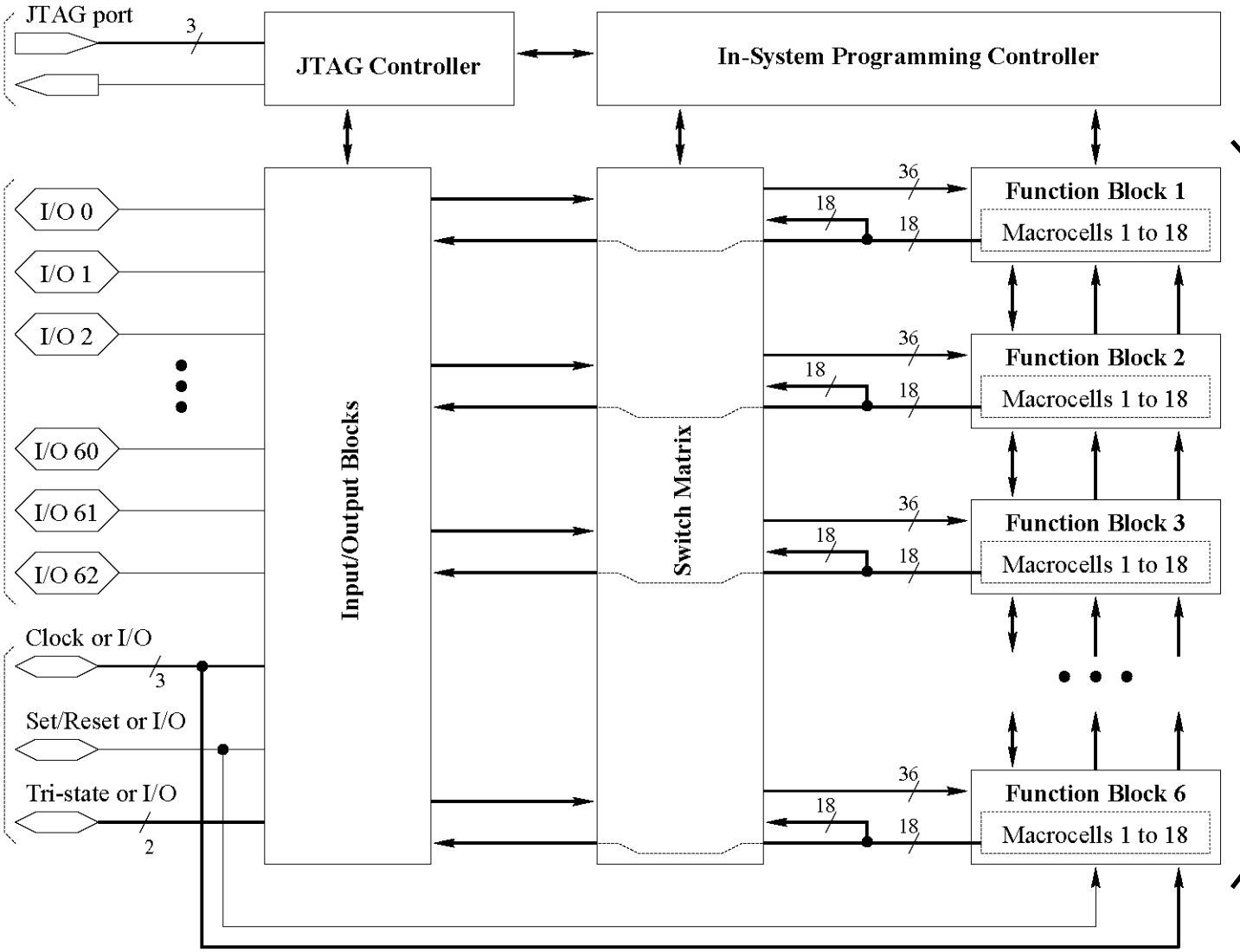
8. CPLDs e Memórias

- CPLDs (5) : arquitectura das CPLDs da família 9500 da Xilinx -

- A arquitectura das CPLDs XC95xx da Xilinx é composta por:
  - um conjunto de **blocos funcionais** e
  - de **blocos de entrada/saída** (IOBs) interligados por
  - uma **matriz de comutação** que encaminha as entradas e saídas para os blocos funcionais.

8. CPLDs e Memórias

- CPLDs (6) : arquitectura das CPLDs da família 9500 da Xilinx -



108 macro-células → XC95108

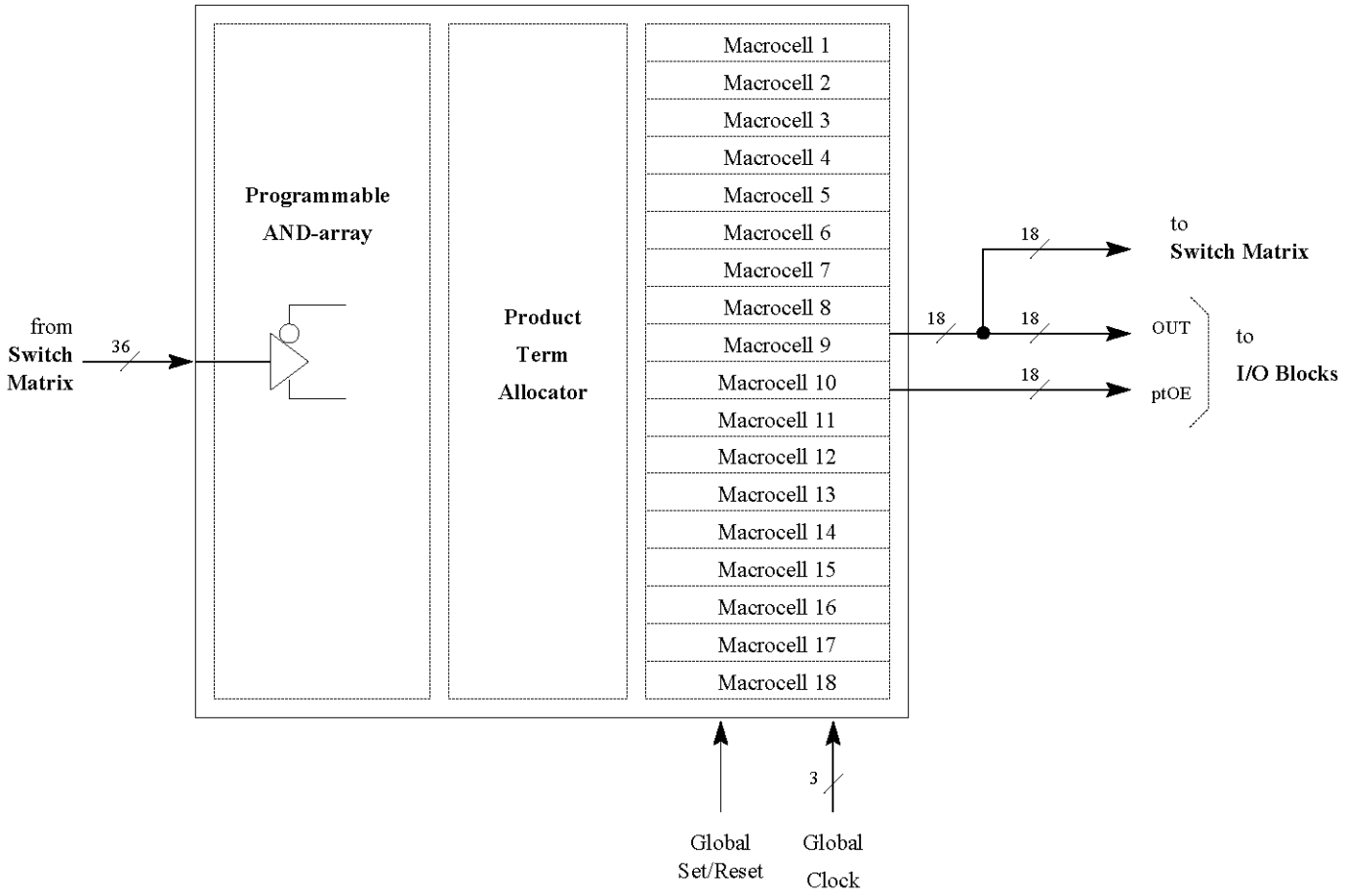
8. CPLDs e Memórias

- CPLDs (7) : bloco funcional das CPLDs da família 9500 da Xilinx -

- Cada **bloco funcional** possui 18 macrocélulas, cada uma implementa uma função combinacional ou registada.
- Entradas de cada bloco: sinais de relógio globais, um sinal de *set/reset* global, sinais de *enable* e até 36 entradas genéricas.
- Cada bloco gera até 18 saídas para a matriz de comutação e/ou IOBs.
- Com as 36 entradas e o complemento dessas 36 entradas pode gerar-se até 90 termos de produto [18 macrocélulas * 5 termos por macrocélula = 90 termos].
- Existem caminhos de realimentação dentro do bloco funcional, permitindo que as saídas desse bloco possam ser usadas como entradas do *array* de ANDs do mesmo bloco.

8. CPLDs e Memórias

- CPLDs (8) : bloco funcional das CPLDs da família 9500 da Xilinx -



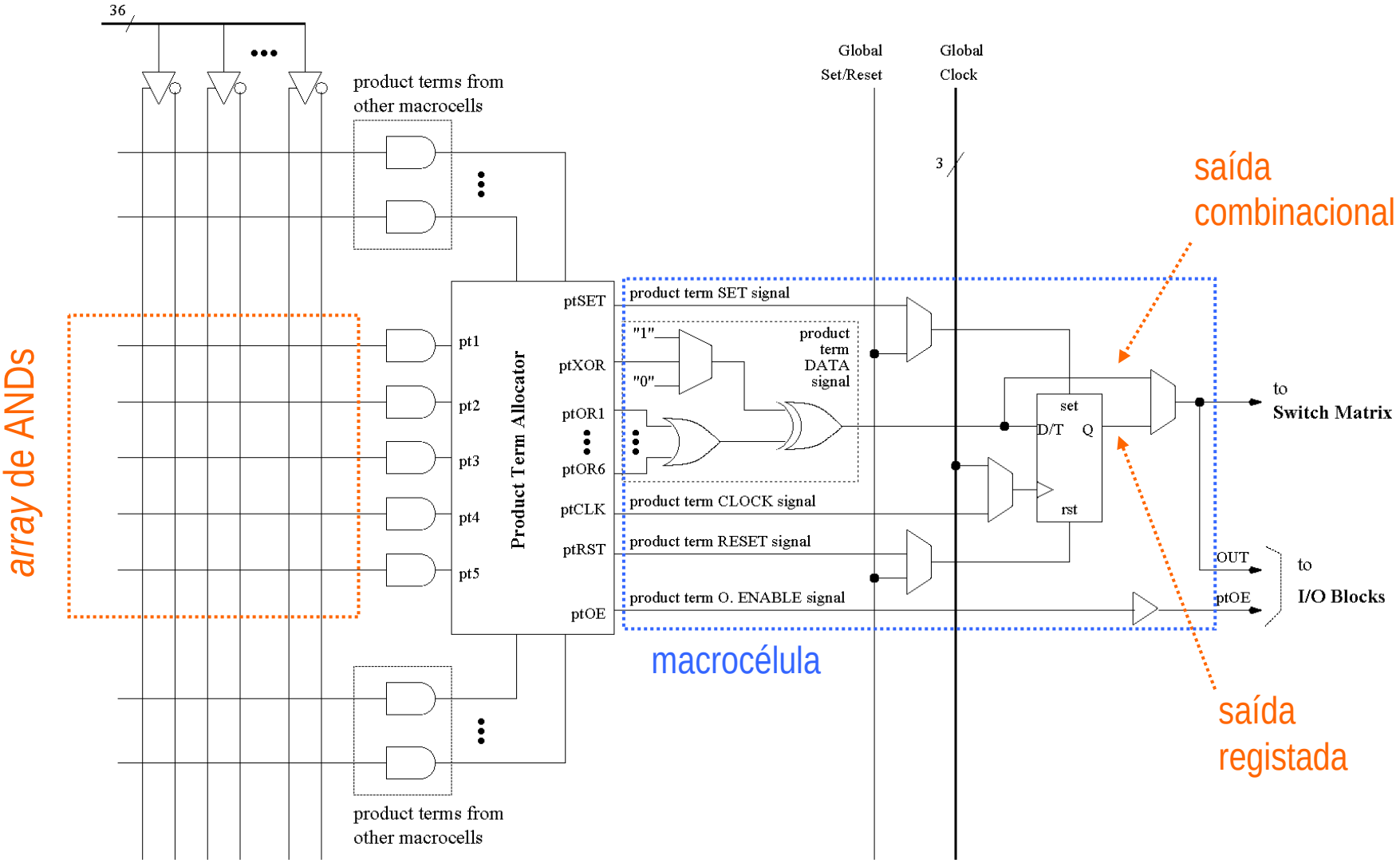
8. CPLDs e Memórias

- CPLDs (9) : macrocélula das CPLDs da família 9500 da Xilinx -

- Cada **macrocélula** pode implementar uma função combinacional ou registada.
- Os contributos para os termos de produto são gerados pelo alocador de termos de produto a partir de:
  - 5 entradas *pt1* a *pt5* provenientes directamente do *array* de ANDs.
  - alguns termos de produto provenientes de outras macrocélulas [1 da célula acima e 1 da célula abaixo, no exemplo a apresentar].
- O alocador de termos de produto gera contributos para:
  - a saída duma função combinacional.
  - a entrada de dados / o sinal de relógio / os sinais de *set* e de *reset* duma função registada.
  - uma saída de *enable*.

8. CPLDs e Memórias

- CPLDs (10) : macrocélula das CPLDs da família 9500 da Xilinx -



8. CPLDs e Memórias

- CPLDs (11) : alocador de produtos das CPLDs da família 9500 da Xilinx -

- O **alocador de termos de produto** controla o modo como os termos de produto são utilizados na implementação de cada função lógica.

- Uma função lógica pode envolver todos os 90 termos de produto, mas se usar apenas 15 termos (5 da macrocélula e 5/5 da macrocélula acima/abaixo) o atraso é mínimo.

- O alocador de termos de produto pode gerar somas de produto parciais a usar nessa ou em outras macrocélulas.

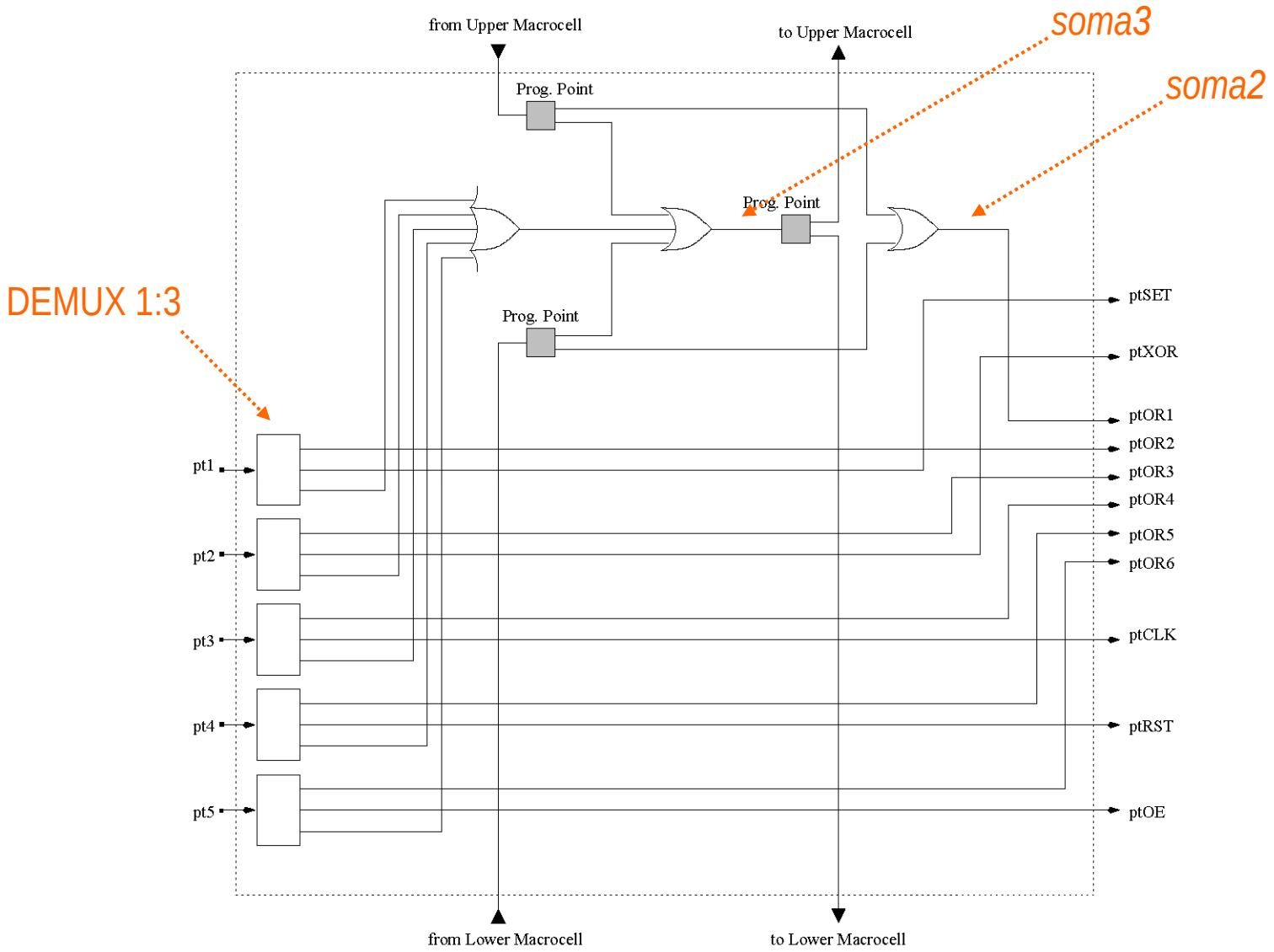
- Por exemplo, o alocador de termos de produto gera 2 sinais:

  - o primeiro sinal é uma soma de produtos parcial, envolvendo 3 termos de produto, a enviar para as macrocélulas vizinhas [sinal *soma3* no próximo slide].

  - o segundo sinal é uma soma de 2 termos de produto, provenientes de macrocélulas vizinhas, que implementa a função lógica da própria macrocélula [sinal *soma2* no próximo slide].

8. CPLDs e Memórias

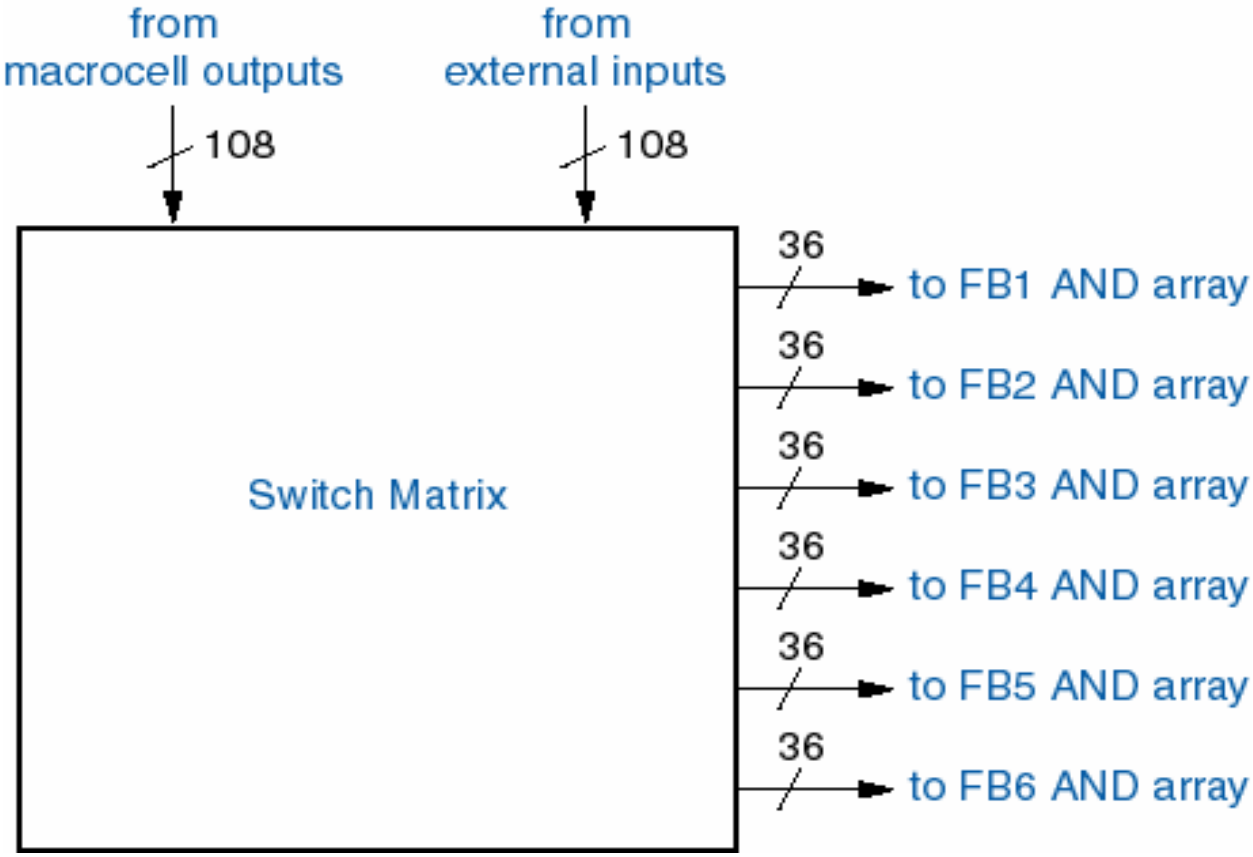
- CPLDs (12) : alocador de produtos das CPLDs da família 9500 da Xilinx -



8. CPLDs e Memórias

- CPLDs (13) : matriz de comutação das CPLDs da família 9500 da Xilinx -

- A **matriz de comutação** disponibiliza caminhos programáveis entre as entradas e as saídas.
- As entradas: são as saídas dos *buffers* de entrada dos IOBs e as 18 saídas de cada FB.
- As saídas: ligam às 36 entradas de cada FB.



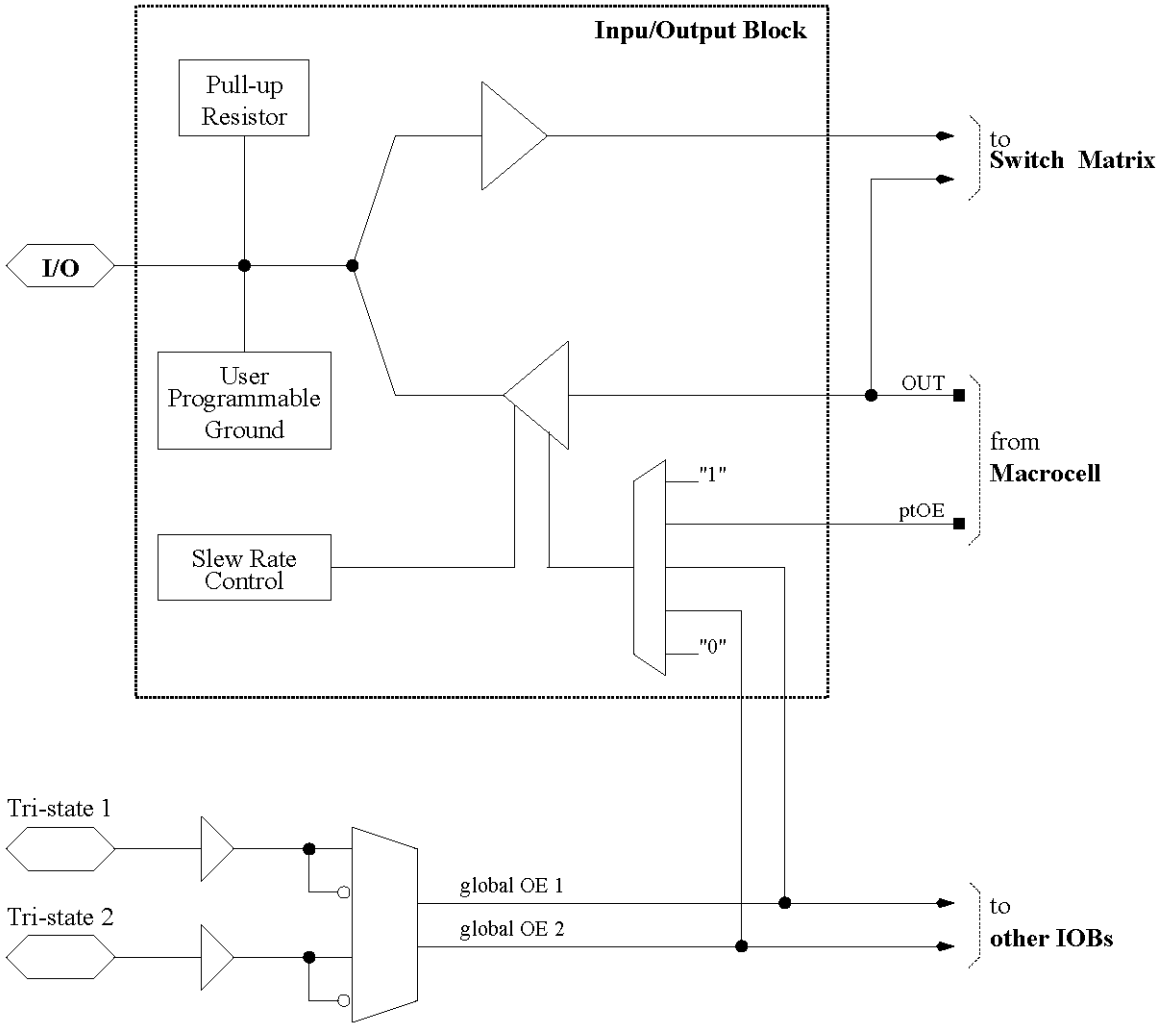
8. CPLDs e Memórias

- CPLDs (14) : bloco de entrada/saída das CPLDs da família 9500 da Xilinx -

- O **bloco de entrada/saída** (IOB) funciona como interface entre a lógica interna e os pinos.
- Cada IOB contém:
  - um *buffer* de entrada.
  - um *buffer tri-state* para saída.
  - lógica para gerar o sinal que controla o *buffer* de saída.
- Este sinal pode ser gerado de várias formas:
  - é o sinal de *output enable* gerado internamente pelo alocador de termos de produto.
  - é um de entre os dois sinais de *output enable* globais/externos (*OE1* ou *OE2*).
  - é um sinal fixo a '0' (*buffer tri-state sempre desligado => pino só de entrada*).
  - é um sinal fixo a '1' (*buffer tri-state sempre ligado => pino só de* saída).

8. CPLDs e Memórias

- CPLDs (15) : bloco de entrada/saída das CPLDs da família 9500 da Xilinx -



8. CPLDs e Memórias

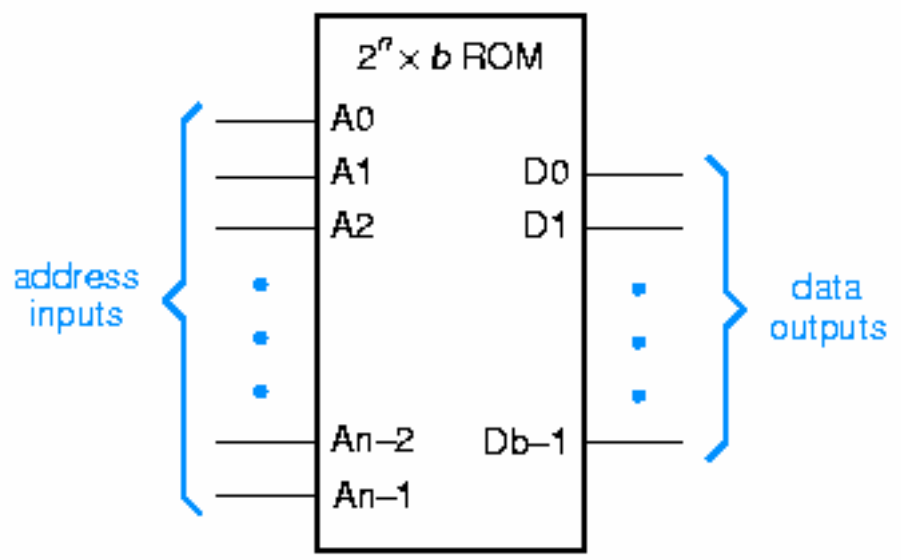
- Memória -

- Qualquer circuito sequencial possui pelo menos um tipo de memória,uma vez que cada *flip-flop* e *latch* guarda um bit de informação.
- Contudo, o termo **memória** é usado para identificar um dispositivo em que os bits estão armazenados de forma estruturada, normalmente em forma de *array* bi-dimensional, em que se acede a uma linha de bits de cada vez.
- O campo de aplicação da memória é vasto e variado:
  - em microprocessadores (uPs);
  - em sistemas baseados em uPs ou microcontroladores, para guardar os dados a processar e/ou as instruções a executar;
  - em dispositivos de armazenamento portáteis: cartões SD/CF, flash drives, leitores MP3
  - em sistemas audio/vídeo (como leitores/gravadores de CD/DVD) para guardar uma parte da informação a processar e assim melhorar o desempenho, etc, etc.
- No CPU dum microprocessador, a ROM pode ser usada para guardar informação que define os procedimentos básicos a executar no arranque do sistema.
- A *cache* dos microprocessadores também é uma memória: –capacidade / +desempenho.
- A **memória principal** é uma memória +capacidade / -desempenho.

8. CPLDs e Memórias

- ROMs (1) -

- Uma memória só de leitura *(ROM - Read-Only Memory)* é um circuito combinacional com **n** entradas e **b** saídas. →
- As entradas definem o endereço de entrada e as saídas os dados de saída.
- Uma ROM pode ser vista como um dispositivo que guarda a tabela de verdade duma função lógica combinacional com **n** entradas e **b** saídas.



8. CPLDs e Memórias

- ROMs (2) -

exemplo duma função

Inputs			Outputs			
A2	A1	A0	D3	D2	D1	D0
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	1
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

no endereço 5 está guardado “0010”

- A tabela de verdade duma função lógica combinacional com 3 entradas e 4 saídas pode ser guardada numa ROM de tamanho 2^3x4.
- Os dados de saída da ROM coincidem com o valor das saídas na linha *(da tabela de verdade)* seleccionada pelos bits de endereço.
- Como uma ROM é um circuito combinacional, não é verdadeiramente uma memória.
- Mas pode pensar-se na ROM como um dispostivo que guarda a informação definida no momento em que ela foi fabricada.
- A ROM é uma memória não-volátil, dado que mantém o conteúdo mesmo quando se desliga a alimentação.

8. CPLDs e Memórias

- ROMs (3) -

- Estrutura interna duma ROM:
  - Depende da tecnologia usada no seu fabrico.
  - Mas é habitual incluir/não-incluir um diodo ou transistor para "programar" um 1/0 numa dada posição da matriz da ROM.
  - A próxima figura ilustra a estrutura duma ROM 128x1, em que a descodificação do endereço se processa em 2 dimensões para reduzir o tamanho do descodificador.
  - A ROM 128x1 está organizada segundo uma matriz bi-dimensional de 8 linhas x 16 colunas.
  - O descodificador usa os 3 bits MS do endereço para seleccionar uma linha.
  - O MUX 16:1 usa os 4 bits LS do endereço para seleccionar uma coluna.

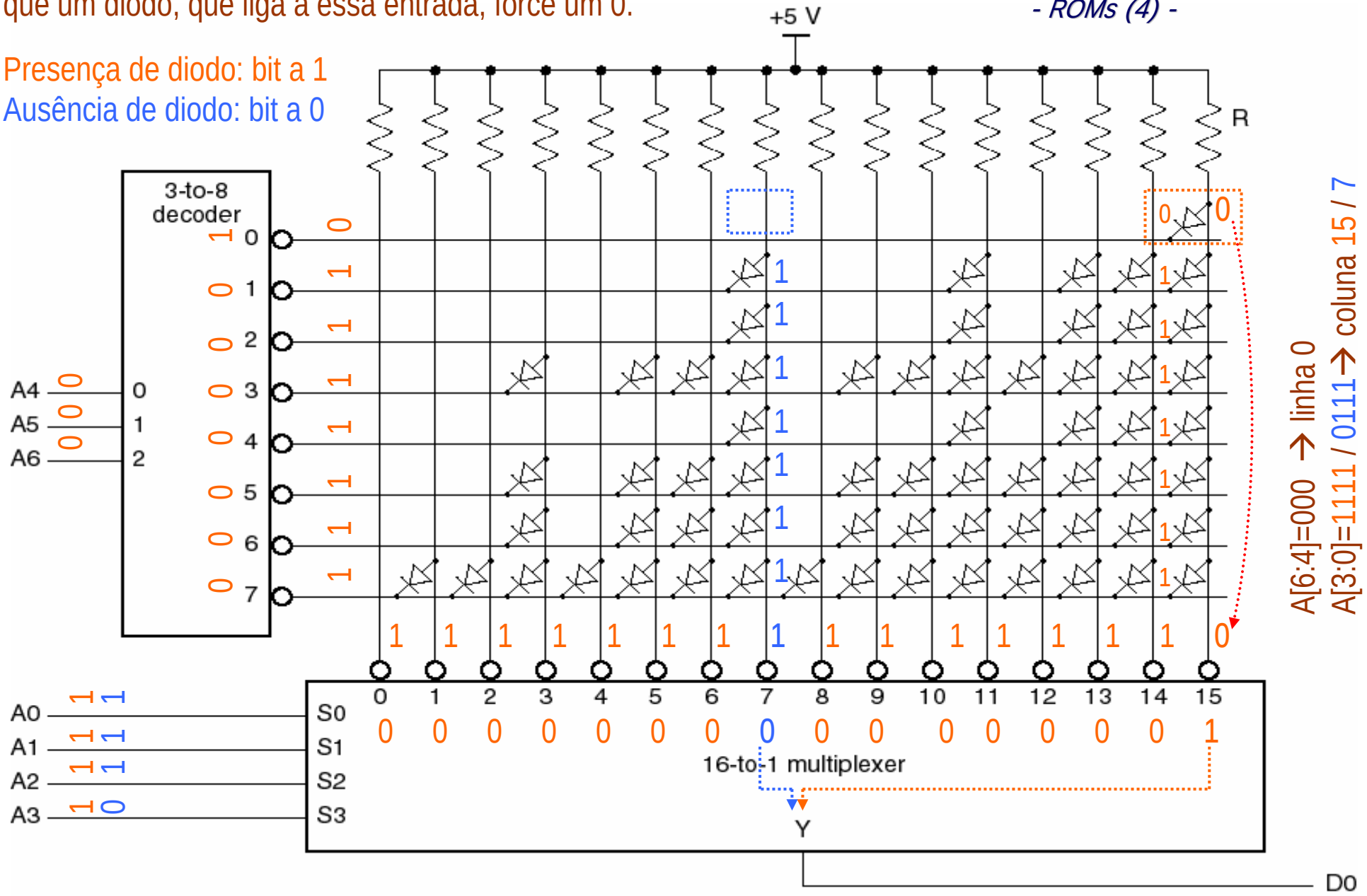
8. CPLDs e Memórias

- ROMs (4) -

Para que uma entrada do MUX seja 0 só é necessário que um diodo, que liga a essa entrada, force um 0.

Presença de diodo: bit a 1

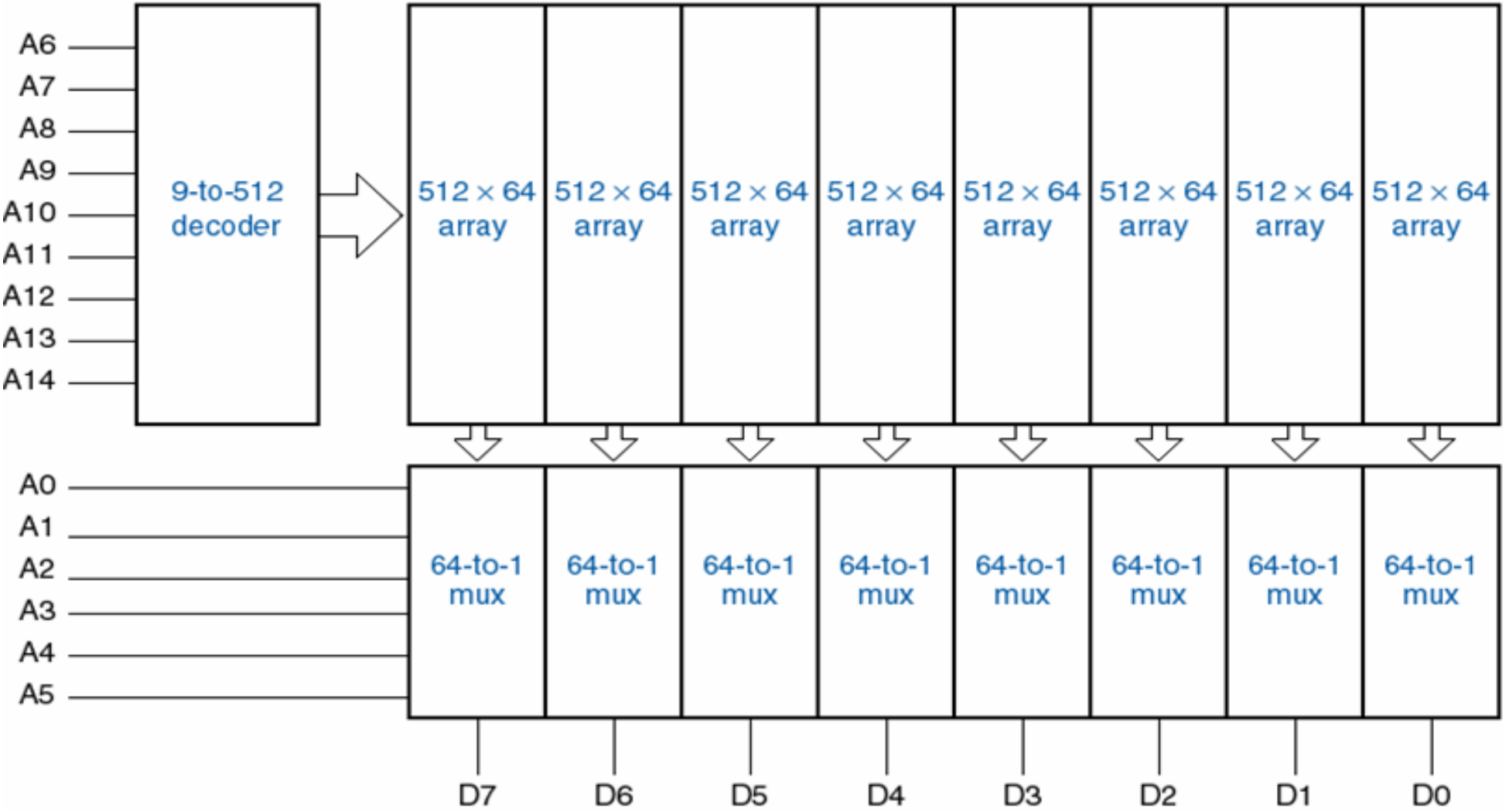
Ausência de diodo: bit a 0



8. CPLDs e Memórias

- ROMs (5) -

Uma ROM de tamanho 32K com múltiplas saídas (D7:D0) possui 2^{15}x8 bits.
Usa 1 descodificador de 9:512, 8 matrizes de 512 linhas x 64 colunas e 8 MUX 64:1.



8. CPLDs e Memórias

- ROMs (6) -

- Actualmente, as ROM são fabricadas num único CI e não com componentes discretos.
- Uma ROM com capacidade de alguns Mbits custa menos de 5€.
- O conteúdo duma ROM pode ser "programado" usando um de vários métodos.
- Para programar o conteúdo duma *mask-programmable ROM*, fornece-se ao fabricante uma lista (ficheiro) que define o padrão de ligações e de não-ligações a efectuar na(s) matriz(es) da ROM. Este processo de fabrico é dispendioso e só se usa para produções em grande escala.
- O processo de programação duma ROM programável (PROM) é idêntico ao da *mask ROM*, excepto que o conteúdo da PROM pode ser gravado usando um programador de PROMs. Uma PROM é fabricada com todos os bits no mesmo valor, normalmente a 1. O programador permite mudar o valor dos bits para 0, nas posições requeridas.
- Uma *erasable programmable ROM* (EPROM) é programável tal como uma PROM, mas o seu conteúdo pode ser apagado e colocado no estado de tudo a 1, usando uma luz ultravioleta.

8. CPLDs e Memórias

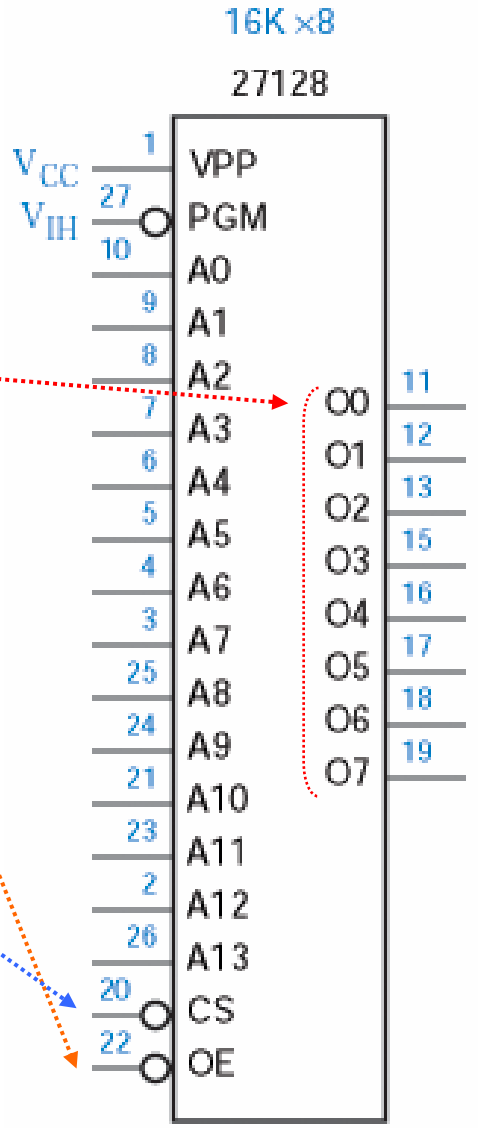
- ROMs (7) -

- Provavelmente a aplicação mais frequente das EPROMs é para guardar o código a executar pelo microprocessador ou microcontrolador dos sistemas embebidos.
- É comum utilizar EPROMs durante o desenvolvimento do código a usar nos sistemas embebidos, dado que esse código vai ser alterado repetidamente durante a fase de depuração.
- Como as ROMs e as PROMs são mais baratas do que as EPROMs similares, quando o desenvolvimento do código estiver concluído, substitui-se a EPROM por uma ROM ou PROM para que a produção do sistema seja menos dispendiosa.
- Uma *electrically erasable programmable ROM* (EEPROM) é idêntica a uma EPROM, excepto que o conteúdo da EEPROM pode ser apagado electricamente (pela aplicação duma tensão).

8. CPLDs e Memórias

- ROMs (8) -

- As EEPROMs nao são alternativa às RAMs porque o tempo de escrita é muito superior e não estão vocacionadas para serem reescritas indefinidamente (apenas milhares de vezes).
- Muitas vezes as saídas da ROM (O7:O0) ligam a um barramento comum, ao qual ligam vários dispositivos com o objectivo de nele escrever, ainda que em instantes distintos.
- Assim, muitas ROMs possuem saídas em *tri-state* e uma entrada OE (*Output Enable*) que deve ser activada para que as saídas apareçam no exterior do *chip*.
- Para facilitar o desenho de certos circuitos em que há multiplas ROMs ligadas ao mesmo barramento, embora apenas uma tenha a saída activa em cada instante, as ROMs possuem uma entrada CS (*Chip Select*).
- Neste caso, para que as saídas em *tri-state* estejam disponíveis no exterior é necessário que as entradas OE e CS estejam ambas activas.

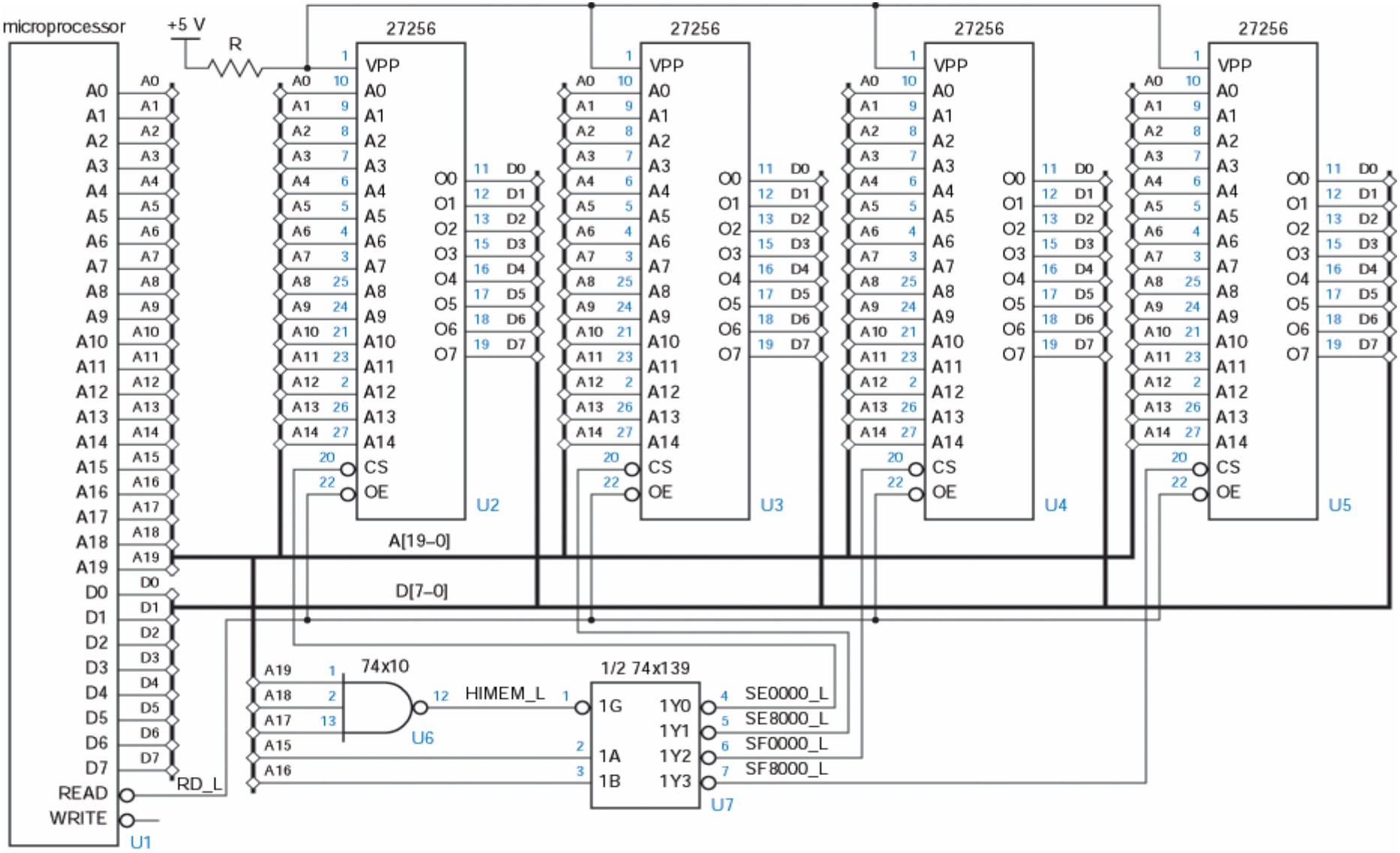


8. CPLDs e Memórias

- ROMs (9) -

Descodificação de endereços para gerar os 4 sinais CS usados na leitura de 4 ROMs 32Kx8 (⇔128Kx8)

HIMEM_L=0 se A19:A17=111 → ROMs estão nos 128K endereços MS
SE0000_L=0 se A16:A15=00 → selecciona a ROM U2



8. CPLDs e Memórias

- ROMs (10) -

Algumas das vantagens em usar ROMs no projecto de circuitos:

- O projecto dos circuitos é rápido e simples.
- O circuito resultante é normalmente mais rápido do que um circuito com vários CIs SSI/MSI/PLDs.
- A funcionalidade implementada pela ROM pode ser alterada facilmente, mudando apenas o conteúdo nela guardado, sem ter que se alterar qualquer lógica no seu exterior.
- O preço das ROMs, por ser um dispositivo estruturado, diminui constantemente.
- A densidade das ROMs aumenta constantemente, alargando o tipo de problemas que se pode resolver com um único *chip*.

8. CPLDs e Memórias

- ROMs (11) -

Algumas das **desvantagens em usar ROMs** no projecto de circuitos:

- Em circuitos simples ou medianamente complexos, uma solução baseada em ROMs pode ser mais cara, consumir mais ou ser mais lenta do que um circuito com vários CIs SSI/MSI/PLDs.
- Para implementar funções com mais de 20 entradas, uma solução baseada em ROMs não é exequível devido à limitação imposta pelo tamanho das ROMs disponíveis. Por exemplo, implementar um somador de 16 bits com ROMs exigiria muitos milhões de bits (2^{32}x16).

8. CPLDs e Memórias

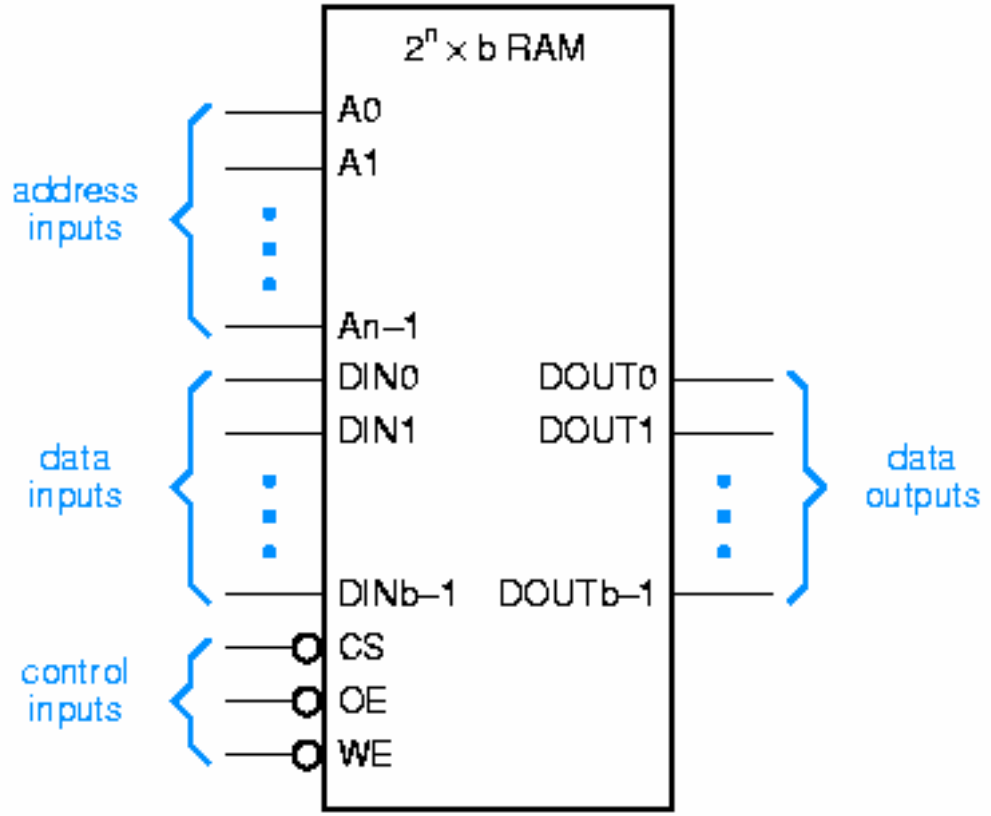
- RAMs (1) : introdução -

- A designação memória de leitura/escrita (*RWM - Read/Write Memory*) aplica-se a *arrays* de memória que permitem que a informação seja guardada e lida em qualquer momento.
- Actualmente, a maior parte das memórias RWM é do tipo RAM.
- Numa memória de acesso aleatório (*RAM - Random-Access Memory*), o tempo que demora a ler ou escrever um bit não depende da sua localização.
- Segundo a definição anterior, as ROMs também são memórias de acesso aleatório, mas a designação “RAM” geralmente só se aplica a RAMs que suportam leitura e escrita.
- Numa memória RAM estática (*SRAM*), o conteúdo escrito numa determinada posição mantém-se enquanto o *chip* estiver alimentado, a não ser que se escreva outro valor nessa posição.
- Uma memória RAM dinâmica (*DRAM*), só mantém o conteúdo guardado em cada posição se ele for reposto *(refreshed)* periodicamente. O refrescamento consegue-se lendo o conteúdo de determinada posição e re-escrevendo-o na mesma posição.

8. CPLDs e Memórias

- RAMs (2) : entradas e saídas duma SRAM -

- A maior parte das RAMs são memórias do tipo volátil. Ou seja, quando se desliga a alimentação perde-se o seu conteúdo.
- Contudo, algumas RAMs mantêm o conteúdo mesmo quando se desliga a alimentação. Neste caso, são memórias do tipo não-volátil.
- Uma RAM de 2^nxb bits possui como entradas um endereço de n bits, b bits de dados e (3) sinais de controlo e como saídas apenas b bits de dados.



8. CPLDs e Memórias

- RAMs (3) : entradas e saídas duma SRAM -

- Os sinais de controlo são semelhantes aos da ROM apresentada, adicionando-se o sinal *WE* que habilita a escrita na RAM → CS, OE e WE.
- Quando se activa as entradas WE e CS, o valor da entrada de dados é escrito na posição da RAM seleccionada pelo endereço.
- Cada posição (ou célula) de memória numa RAM estática comporta-se como uma *latch* D, e não como um *flip-flop* D sensível às transições do relógio.
- Ou seja, sempre que WE e CS estiverem activos, a *latch* associada com a posição seleccionada está aberta: a saída segue a entrada de dados.
- O valor que fica guardado na posição em causa é aquele que estiver pesente na saída de dados da *latch* quando ela fecha.

8. CPLDs e Memórias

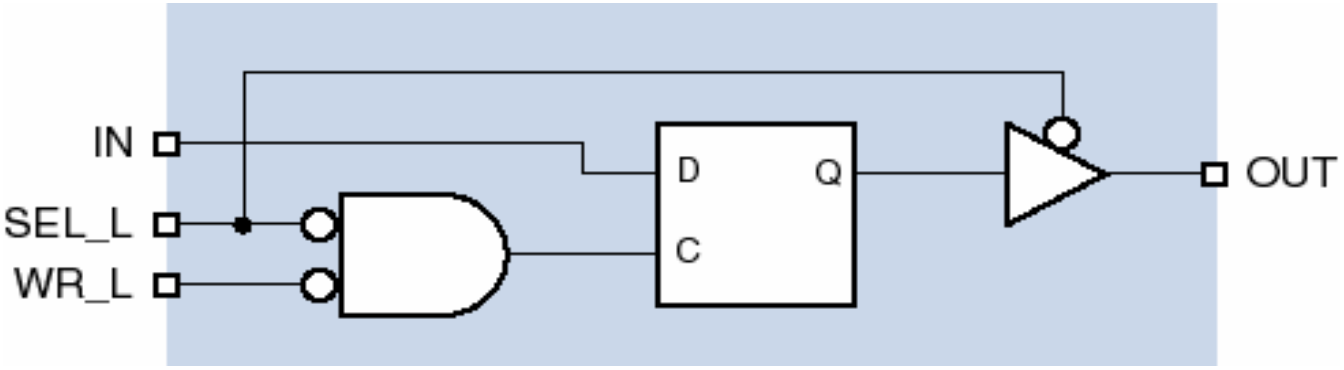
- RAMs (4) : entradas e saídas duma SRAM -

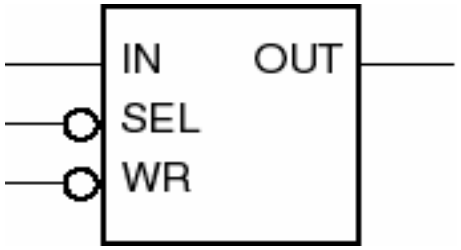
- Normalmente, as RAMs estáticas permitem apenas 2 tipos de acesso: leitura e escrita.
- Para efectuar uma leitura:
  - Coloca-se um endereço na entrada de endereço, mantendo as entradas de controlo CS e OE activas.
  - O valor guardado na *latch* associada com a posição seleccionada é colocado na saída de dados (DOUT na RAM ilustrada atrás).
- Para efectuar uma escrita:
  - Coloca-se um endereço na entrada de endereço e um valor (a escrever) na entrada de dados DIN.
  - Após isso, activam-se as entradas de controlo CS e WE.
  - A *latch* associada com a posição seleccionada abre e o valor presente em DIN é guardado na saída dessa *latch*.

8. CPLDs e Memórias

- RAMs (5) : organização interna duma SRAM -

- A funcionalidade de cada célula da memória SRAM é a mesma que o circuito em baixo:
  - O comportamento é o de uma *latch* D e ***não*** o de um *flip-flop* D sensível às transições.
  - Quando SEL_L estiver activo, o valor guardado na *latch* é colocado na saída OUT.
  - Quando SEL_L e WR_L estiverem ambos activos, a *latch* abre e o valor presente em IN é guardado na *latch*.
- Numa operação de escrita é preciso garantir que:
  - O endereço está estável antes da ordem de escrita.
  - Os dados a escrever ficam estáveis antes de terminar a operação de escrita.





8. CPLDs e Memórias

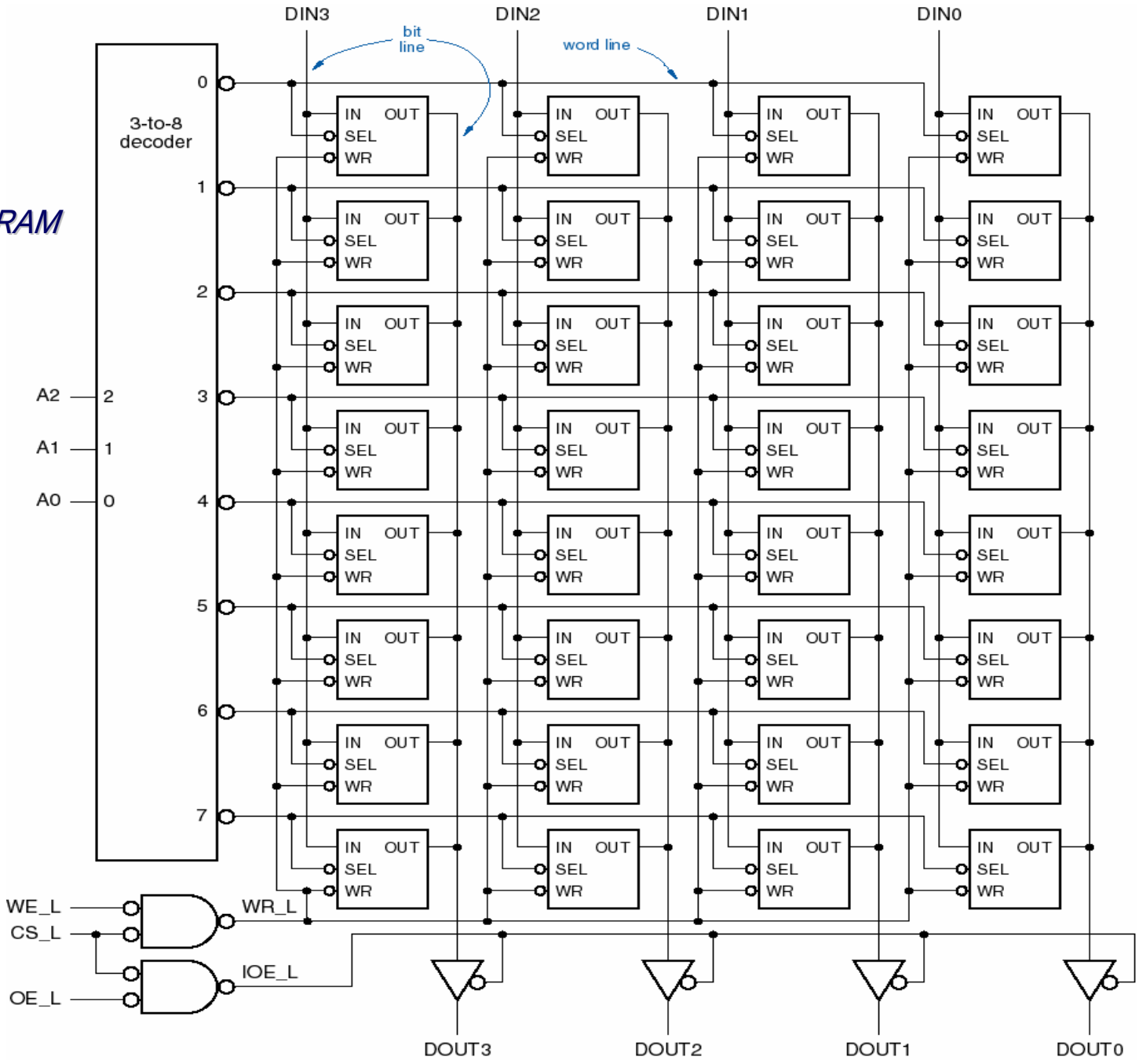
- RAMs (6) -

organização interna duma SRAM

- Combinando várias células de memória como a anterior e juntando alguma lógica de controlo, obtém-se uma SRAM completa, por exemplo com tamanho 8x4 →

- Sinais de controlo:
  - *Chip select* (CS_L)
  - *Output enable* (OE_L)
  - *Write enable* (WE_L)

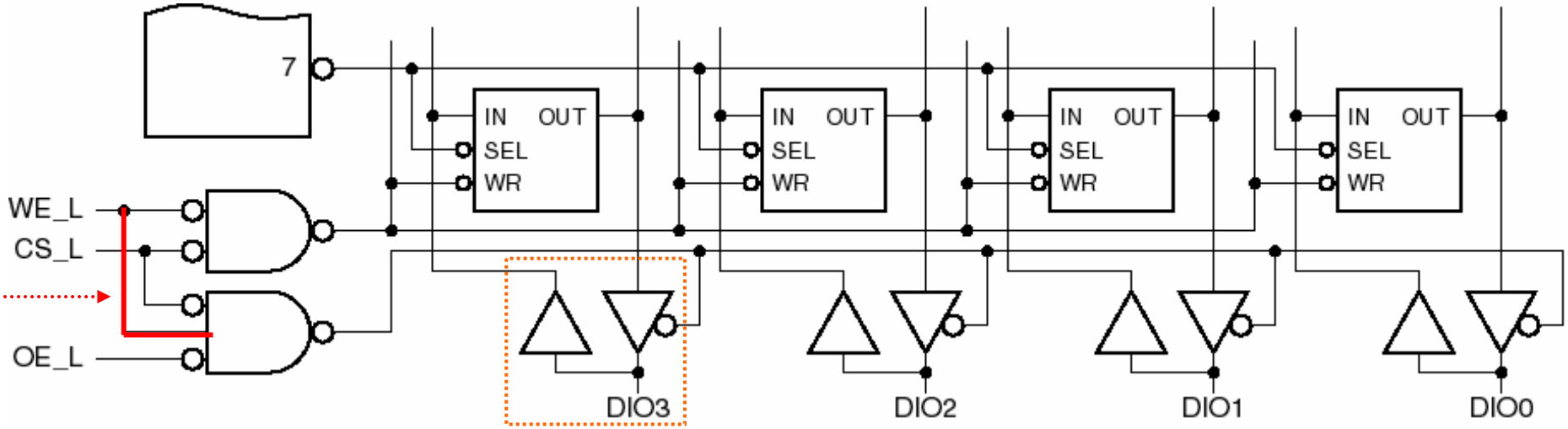
- O descoficador 3:8 activa um (dos 8 sinais) SEL de modo a habilitar a leitura/escrita de/em todas as células da linha correspondente ao endereço A2:A0 aplicado.



8. CPLDs e Memórias

- RAMs (7) : organização interna duma SRAM -

- Para que a entrada e saída dos dados da SRAM se faça pelos mesmos pinos (bi-direccionais) a organização interna apenas tem que ser alterada ligeiramente:



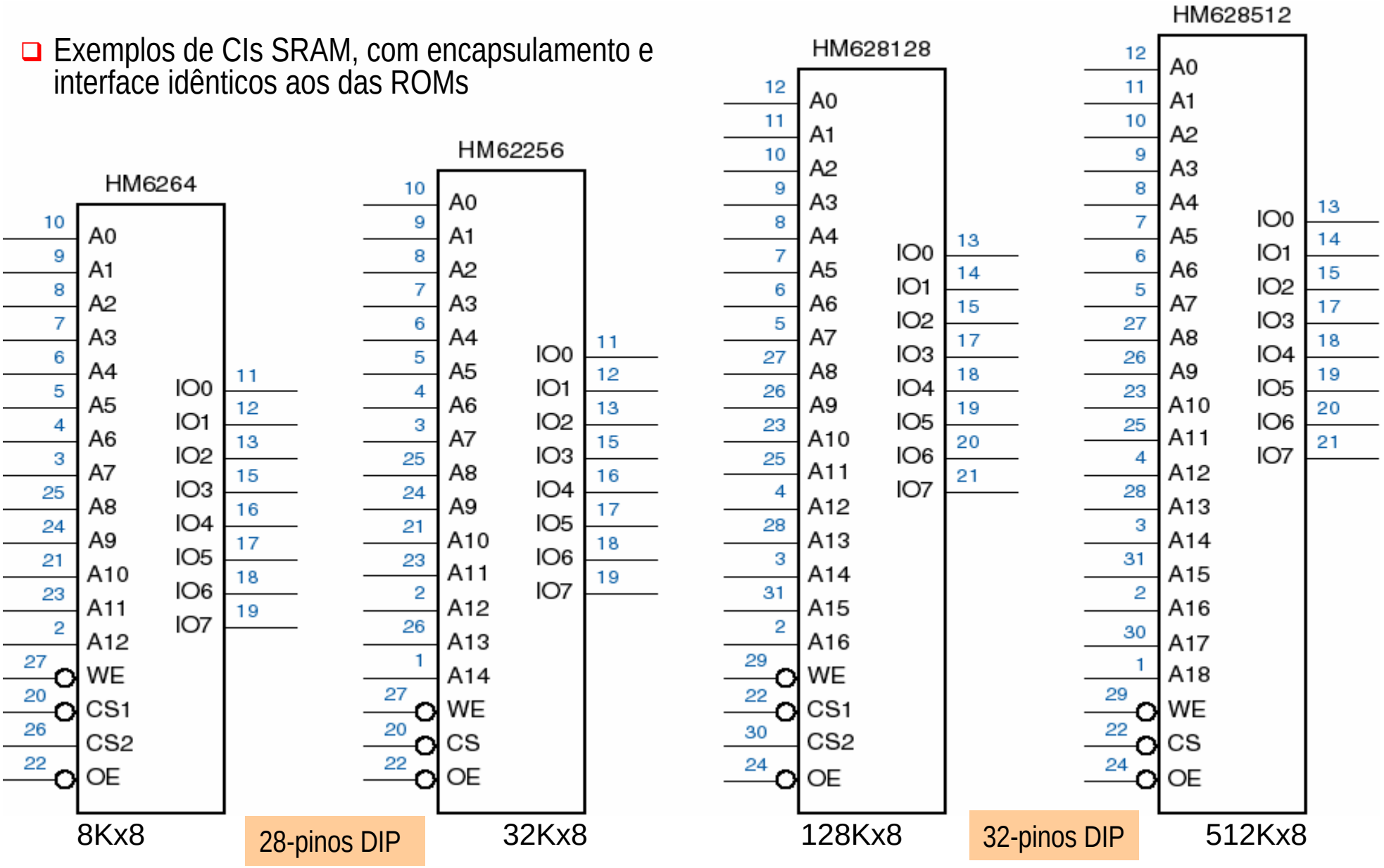
- Qual o interesse em usar os mesmos pinos para os dados a ler e a escrever?
  - Para reduzir o número de pinos, útil em memórias de grande capacidade.
  - Porque a maior aplicação das memórias é na ligação a um microprocessor através de barramentos, sendo o barramento de dados normalmente bi-direccional.

para evitar ler e escrever em simultâneo

8. CPLDs e Memórias

- RAMs (8) : CIs com SRAM -

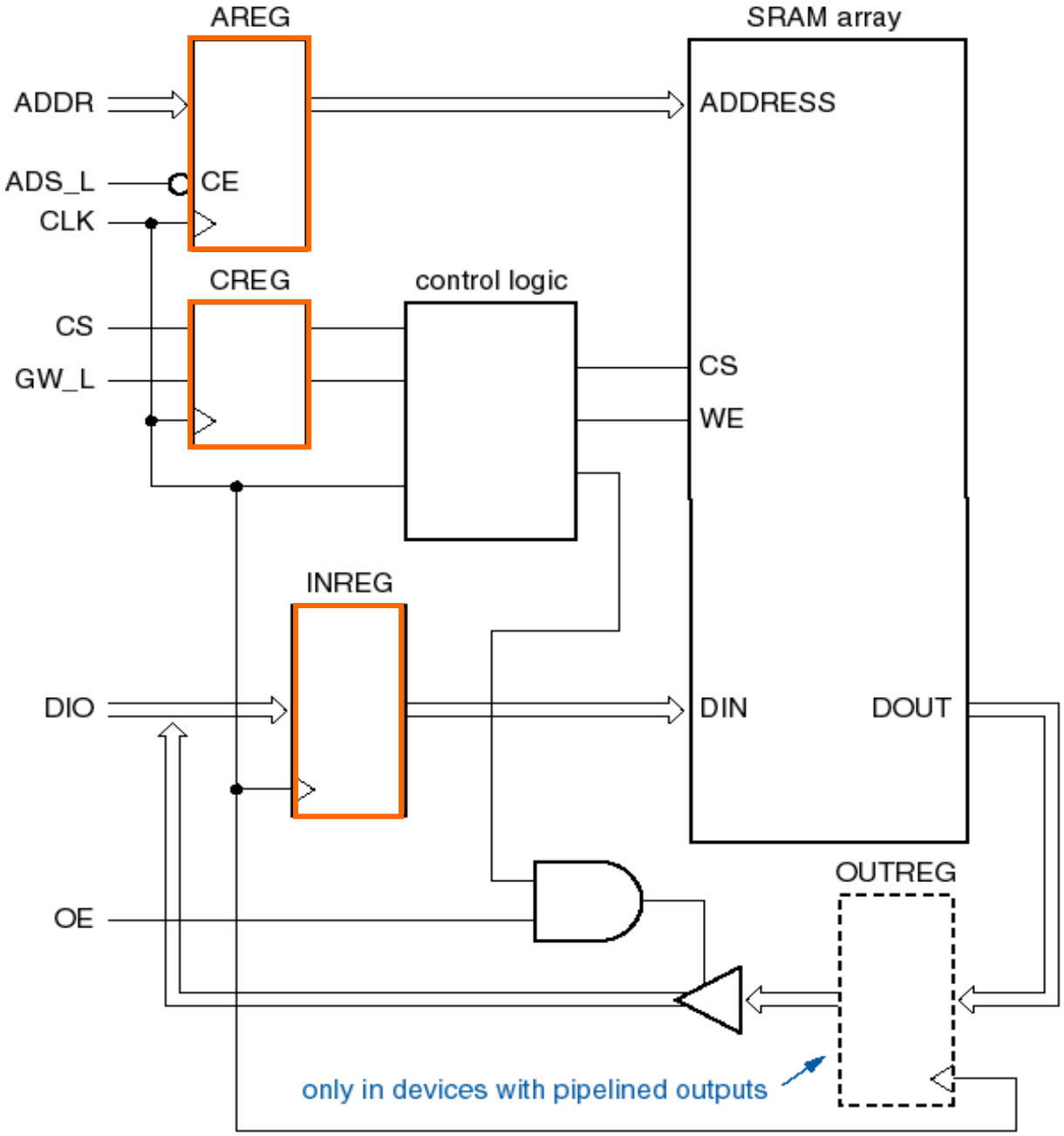
- Exemplos de CIs SRAM, com encapsulamento e interface idênticos aos das ROMs



8. CPLDs e Memórias

- RAMs (9) : SRAMs síncronas -

- As memórias do tipo SRAM síncrona continuam a ter células baseadas numa *latch* mas usam endereços, sinais de controlo e dados registados.
- Na SRAM ao lado, o endereço ADDR é registado em AREG, os sinais de controlo CS/GW_L são registados em CREG e os dados de entrada DIO registados em INREG.
- Registar o endereço e os sinais de controlo facilita a utilização da SRAM em sistemas síncronos que operam com frequências elevadas.



8. CPLDs e Memórias

- RAMs (10) : DRAM -

- A implementação da *latch* D usada nas células da SRAM emprega 4 portas lógicas discretas ou 4 a 6 transistores.
- A memória RAM dinâmica (*DRAM*) surge como forma de diminuir a quantidade de lógica usada por cada célula e conseguir assim uma maior capacidade com o mesmo espaço.
- A célula base da memória DRAM guarda a informação num condensador, que é acedido através dum único transistor.
- Ao fim de alguns milisegundos a informação guardada no condensador precisa ser reposta para que um “1” não passe a “0”. Para repor (refrescar) a informação guardada, lê-se a célula e de seguida ela é re-escrita com a mesma informação.
- Para reduzir o tempo usado no refrescamento, o processo refresca uma linha da matriz de cada vez.
- As DRAMs possuem normalmente várias matrizes para explorar o paralelismo de funcionamento, aumentando o débito de informação a escrever/ler.

8. CPLDs e Memórias

- RAMs (11) : organização interna duma DRAM -

DRAM de 64Kx1 bits organizada segundo uma matriz de 256x256 células.

O endereço de 16 bits é aplicado em **A7:A0** em 2 etapas**,** sendo as metades registadas no bordo descendente de **RAS_L** e **CAS_L**.

ordem para registar a metade do endereço que corresponde à linha da matriz →

ordem para registar a coluna →