

Circuitos combinatórios

Sistemas Digitais

Pedro Salgueiro
pds@uevora.pt

Sumário

- Somador
- Comparador
- Descodificador
- Multiplexador
- Desmultiplexador
- Codificador

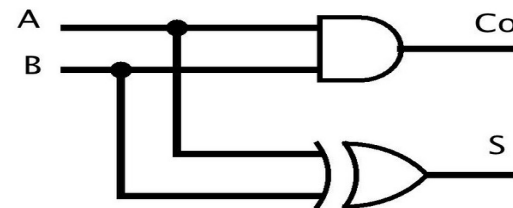
Semi-somador

- **Função**
 - Somar dois algarismos binários
- **Síntese**
 - Entradas: 2
 - Saídas: 2
 - Porque pode produzir transporte (*carry*)
- **Expressão algébrica**
 - $S = A \oplus B$
 - $C_0 = A B$

– Tabela de verdade

A	B	S	C_0
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

– Logigrama



Somador completo

- **Função**

- Somar dois algarismos binários com transporte

- **Síntese**

- Entradas: 3
- Saídas: 2

- **Expressão algébrica**

- $S = A \oplus B \oplus C$
- $C_0 = A B + C_i (A \oplus B)$

- **Tabela de verdade**

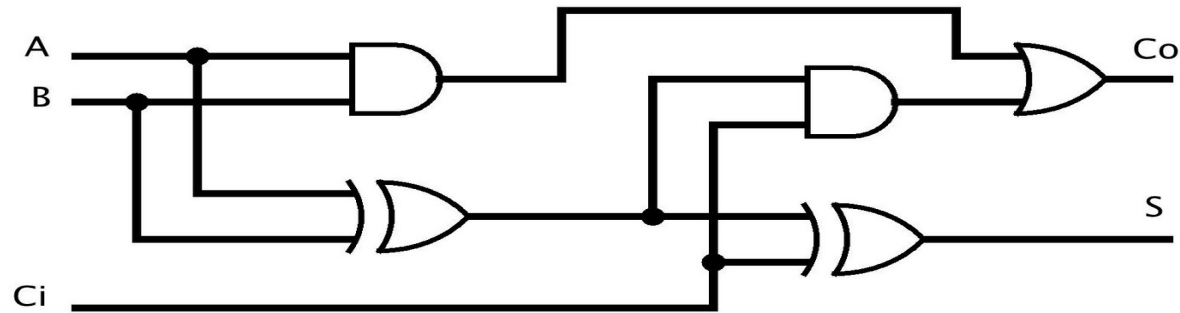
A	B	C_i	S	C_0
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

- **Logigrama**

- Pode ser construído com 2 semi-somadores e 1 porta OR

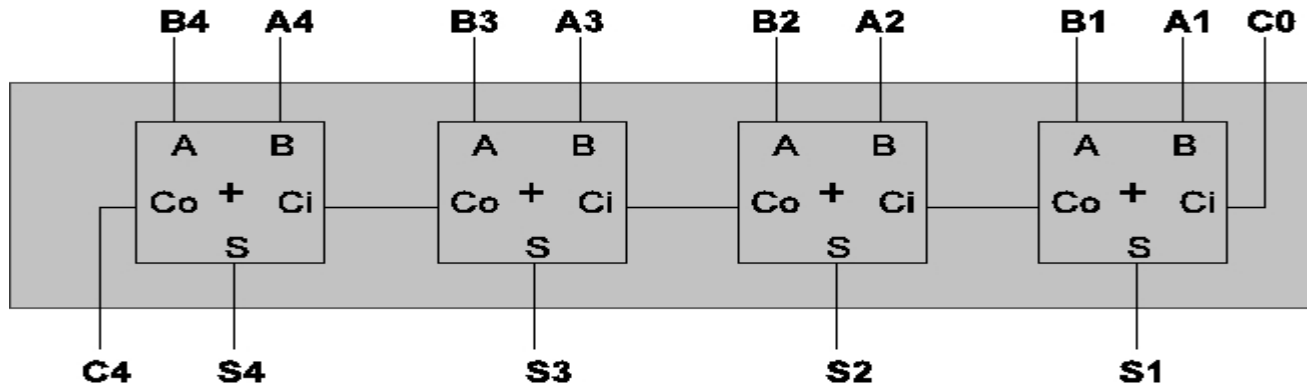
Somador completo

- Logigrama



Somador de 4 bits

- Como construir?
 - A partir de 4 somadores completos



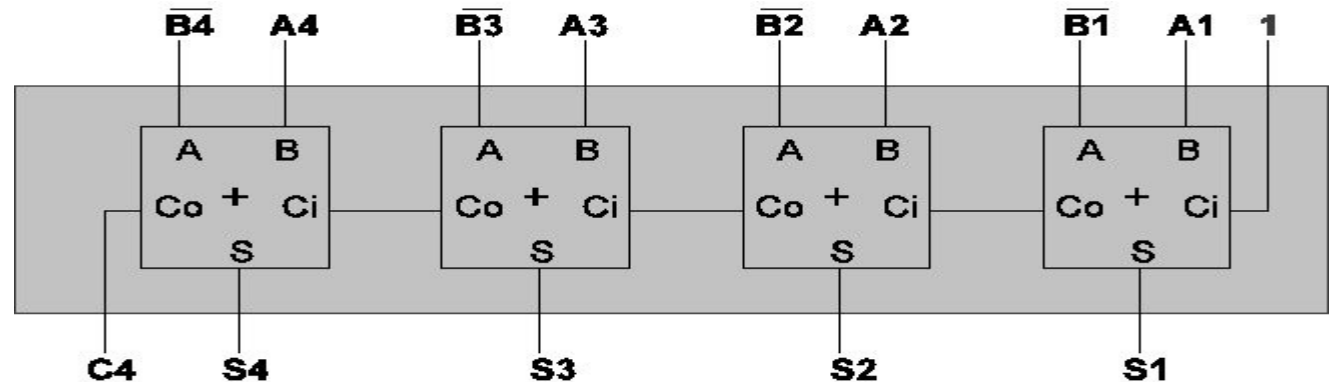
- Circuito integrado
 - TTL 7483

Somador de 8 bits

- Como construir?
 - Utilizar 2 integrados 7483
 - A: 4 bits menos significativos
 - B: 4 bits mais significativos
 - Ligar o C4 do integrado A ao C0 do integrado B

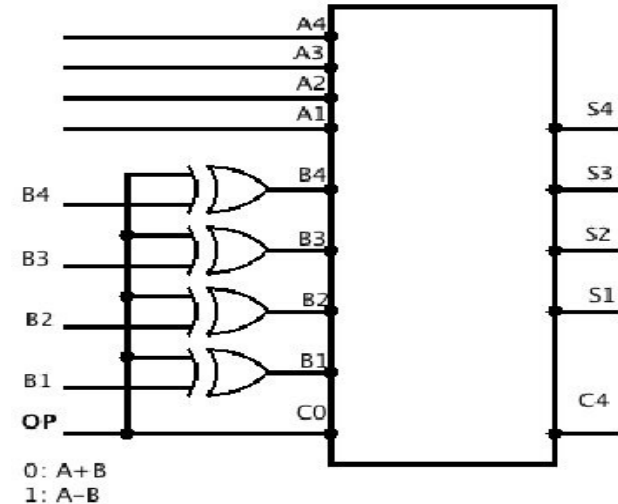
Subtrator

- Como construir?
 - $A - B = A + (-B)$
 - Como obter $-B$?
 - Representação C2: negar B bit a bit e somar 1
- Subtrator 4 bits
 - Utilizar 1 integrado 7483 + 4 NOT



Subtrator/Somador

- Como construir?
 - Utilizar um somador completo
 - Entrada OP: indica a operação a realizar
 - 0: soma
 - 1 subtração
- Somador/subtrator 4 bits
 - Utilizar 1 integrado 7482 + 4 XOR



Overflow

- Quando acontece?
 - Sempre que o transporte do último bit(para o exterior) é diferente do transporte do bit anterior
- Como construir?
 - Usar uma porta XOR

Comparador simples

- **Função**
 - Comparar 2 algarismos binários
 - menor, maior, igual
- **Síntese**
 - Entradas: 2
 - Saídas: 2

– Tabela de verdade

A	B	X	Y
0	0	0	0
0	1	0	1
1	0	1	0
1	1	0	0

Comparador 1 bits

- **Função**
 - Comparar 2 bits
- **Síntese**
 - Entradas: 4
 - Saídas: 2

- **Tabela de verdade**

X_{n+1}	Y_{n+1}	A	B	X_n	Y_n
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	0	0
0	1	x	x	0	1
1	0	x	x	1	0
1	1	x	x	x	x

- Mapa de Karnaugh

X_n

$AB \backslash \begin{matrix} X_{n+1} & Y_{n+1} \end{matrix}$	00	01	11	10
00	0	0	x	1
01	0	0	x	1
11	0	0	x	1
10	1	0	x	1

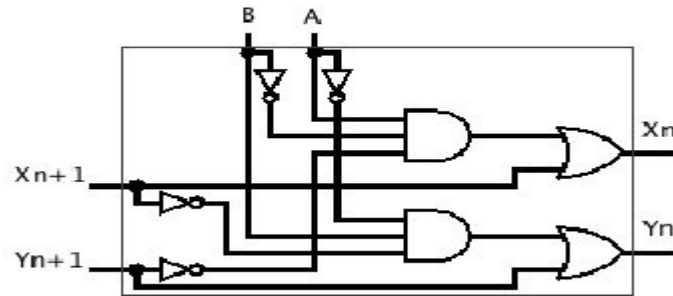
Y_n

$AB \backslash \begin{matrix} X_{n+1} & Y_{n+1} \end{matrix}$	00	01	11	10
00	0	1	x	0
01	1	1	x	0
11	0	1	x	0
10	0	1	x	0

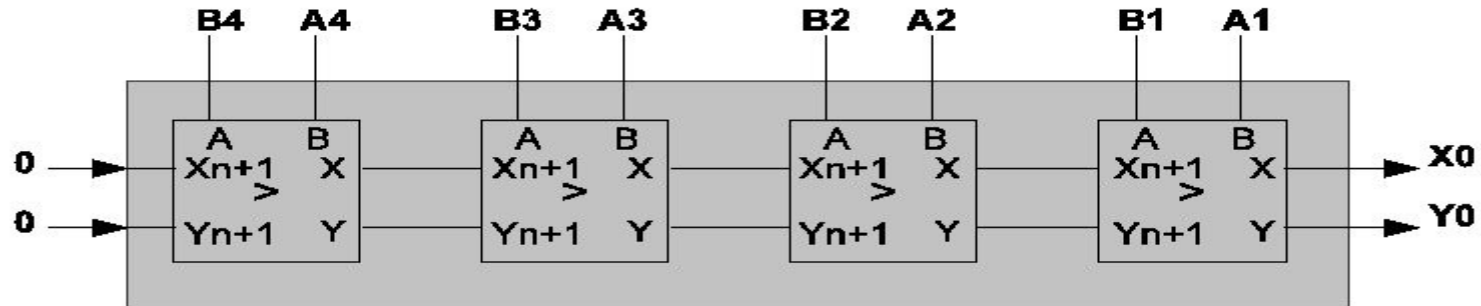
- Expressão algébrica

- $X_n = X_{n+1} + A \bar{B} \bar{Y}_{n+1}$
- $Y_n = Y_{n+1} + \bar{A} B \bar{X}_{n+1}$

- Logigrama



- Comparador 4 bits
 - Como construir?
 - A partir de 4 comparadores



- Circuito integrado
 - TTL 7485
 - 3 saídas: $A=B$, $A > B$ e $A < B$

- **Descodificador**
 - **Função**
 - Identificar as palavras de um código
 - **Síntese**
 - Entradas: comprimento do código
 - Saídas: nº de palavras do código
 - Fica ativa apenas a saída que corresponder à palavra de código presente nas entradas
 - **Características**
 - Em cada instante, apenas uma das saídas está ativa

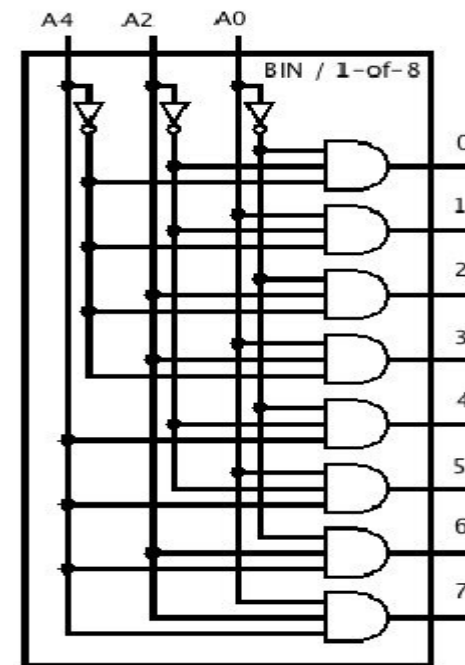
- **Exemplos**
 - **Descodificador números binários de n bits**
 - Entradas: n
 - São aplicadas as palavras de código binário natural
 - Saídas: 2^n
 - Apenas fica ativa a saída que corresponde ao CBN presente à entrada
 - **Descodificador BCD/decimal**
 - Entradas: 4
 - São aplicadas as palavras do código BCD
 - Saídas: 10
 - Apenas fica ativa a saída correspondente ao número representado à entrada

Descodificador binário de 3 bits

- Características
 - Entradas: 3
 - Saídas: 8
- Tabela de verdade

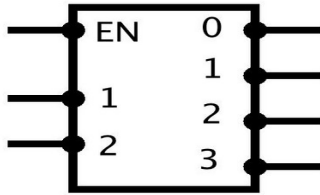
A	B	C	S ₀	S ₁	S ₂	S ₃	S ₄	S ₅	S ₆	S ₇
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

- Logigrama



Circuito integrado

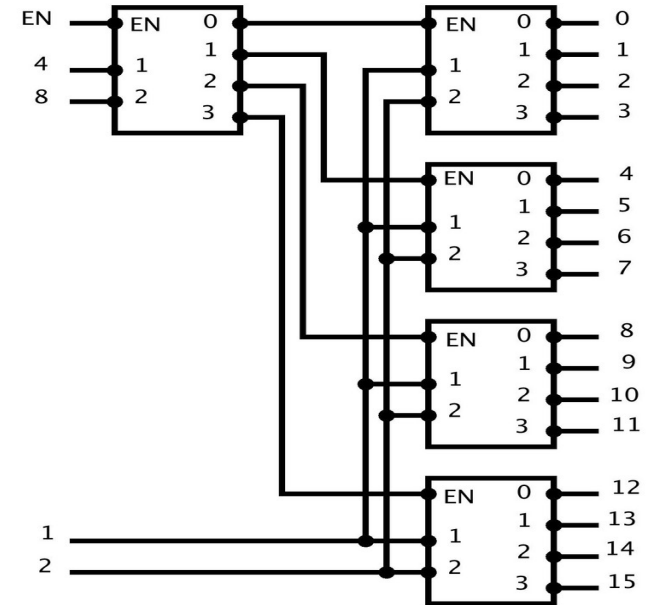
- Os CI têm uma entrada ***Enable***
 - Ativada: o decodificador funciona normalmente
 - Desativada: todas as saídas são desativadas
- Decodificador de 2 bits



Expansão de descodificadores

- Como construir?
 - Utilizar a entrada **Enable** do CI
- Exemplo
 - Construir um descodificador de 4 bits a partir de descodificadores de 2 bits
 - Quantos descodificadores são necessários?
 - 4 para 16 palavras ($16 = 4 \times 4$)
 - 1 para seleccionar o descodificador correto

- Circuito



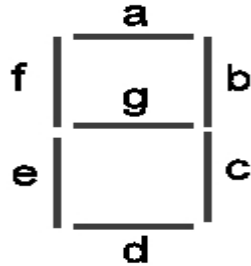
Descodificador BCD/decimal

- **Características**
 - Entradas: 4
 - Saídas: 10
- **Tabela de verdade**
 - As saídas não preenchidas correspondem a '0's
- **CI**
 - **TTL-7442**

[illegible]

Decodificador BCD/7 segmentos

- Display 7 segmentos



- Características

- Entradas: 4
- Saídas: 7

- CI

- TTL-7447
- TTL-7448

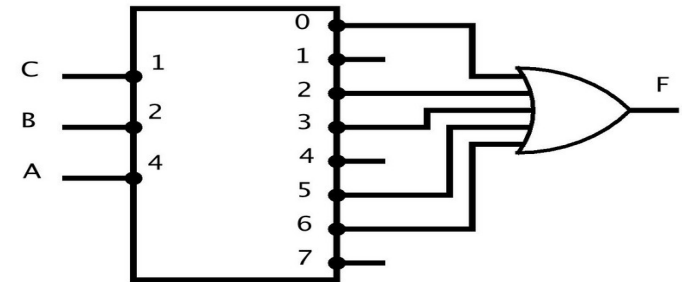
- Tabela de verdade

- As saídas não preenchidas correspondem a '0's

A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	
0	0	0	1		1	1				
0	0	1	0	1	1		1	1		1
0	0	1	1	1	1	1	1			1
0	1	0	0		1	1			1	1
0	1	0	1	1		1	1		1	1
0	1	1	0	1		1	1	1	1	1
0	1	1	1	1	1	1				
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1			1	1
outras										

Descodificadores e funções lógicas

- Qualquer função pode ser implementada com um descodificador e uma porta lógica
 - descodificador implementa os **mintermos** da função
 - porta OR implementa a **soma** dos mintermos
- Exemplo**
 - $F(A,B,C) = \sum m(0,2,3,5,6)$



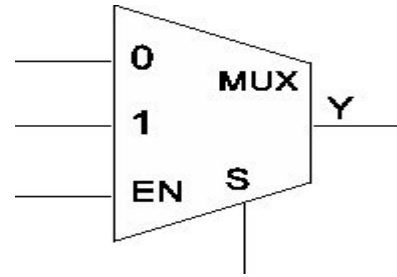
- **Função**
 - Selecionar uma entrada de acordo com a palavra de controlo/seleção
- **Síntese**
 - Entradas de dados: N
 - Entradas de controlo (S): $\log N$
 - Saídas (Y): 1
- **Características**
 - Para distinguir se na saída está uma palavra da entrada ou não, é necessária uma entrada de *Enable*
- **CI**
 - 16 para 1: TTL 74LS150
 - 8 para 1: TTL 74LS151
 - 2 x 4 para 1: TTL 74LS153
 - 4 x 2 para 1: TTL 74LS157

Multiplexador (MUX)

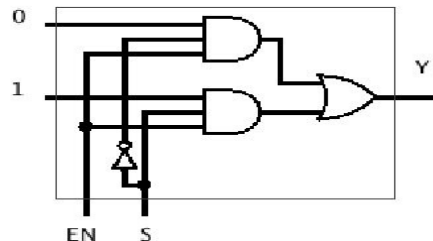
- Tabela de verdade

1	0	S	EN	Y
x	x	x	0	0
x	0	0	1	0
x	1	0	1	1
0	x	1	1	0
1	x	1	1	1

- Símbolo lógico



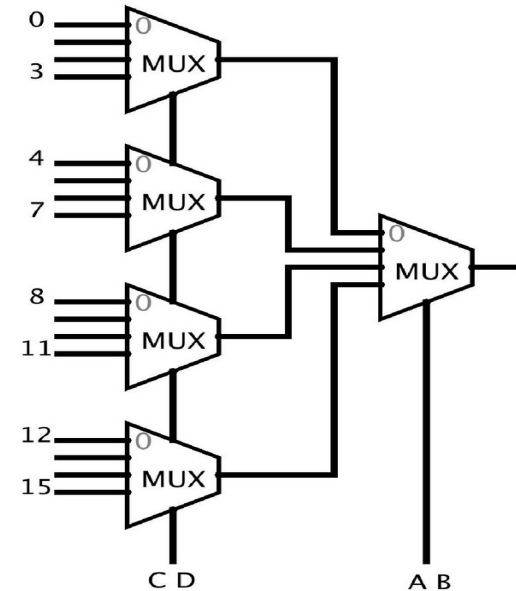
- Logigrama



Expansão de multiplexadores

- Como construir?
 - Em camadas sucessivas numa estrutura em árvore
- Exemplo
 - Construir um multiplexador de 16 para 1 a partir de multiplexadores de 4 para 1

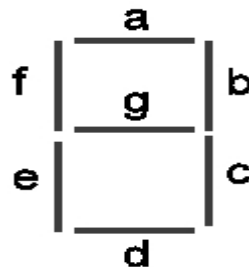
- Circuito



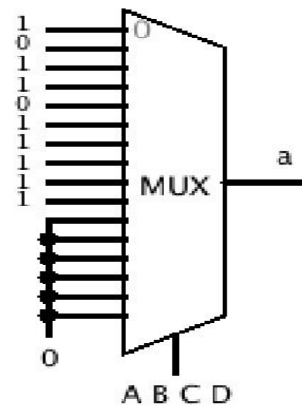
Multiplexador e expressões lógicas

- Qualquer função pode ser implementada com um multiplexador
- Exemplo
 - Saída do segmento **a** do decodificador BCD/7 segmentos

A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	
0	0	0	1		1	1				
0	0	1	0	1	1		1	1		1
0	0	1	1	1	1	1	1			1
0	1	0	0		1	1			1	1
0	1	0	1	1		1	1		1	1
0	1	1	0	1		1	1	1	1	1
0	1	1	1	1	1	1	1			
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1			1	1
outras										



- Símbolo lógico



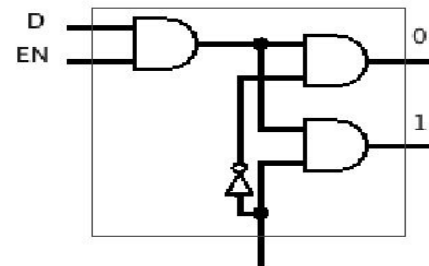
- Função
 - De acordo com uma palavra de controlo, coloca numa saída o valor que está à entrada
- Síntese
 - Entradas de dados (D): 1
 - Entradas de controlo (S): $\log N$
 - Saídas: N
- Características
 - Para distinguir se à saída está uma palavra de entrada ou não, é necessária uma entrada Enable
- CI
 - 2 x 1 de 4: TTL 74LS139

DEMUX 2 saídas

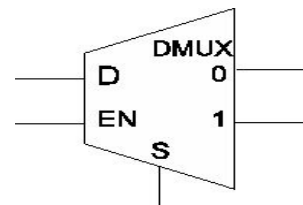
- Tabela de verdade

D	S	EN	0	1
x	x	0	0	0
0	0	1	0	0
1	0	1	1	0
0	1	1	0	0
1	1	1	0	1

- Logigrama



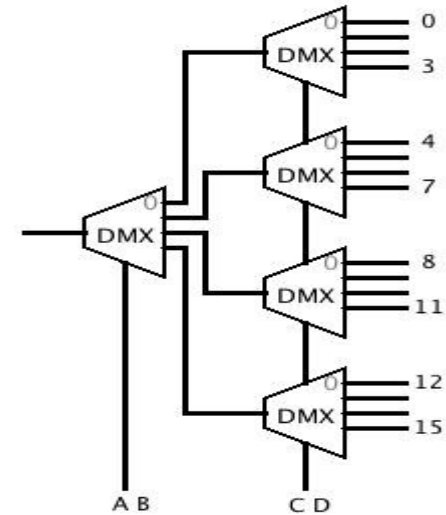
- Símbolo lógico



Expansão de desmultiplexadores

- Como construir?
 - Em camadas sucessivas numa estrutura em árvore
- Exemplo
 - Construir um desmultiplexador 1 de 16 a partir de desmultiplexadores 1 de 4

- Circuito



Desmultiplexador/descodificador

- O desmultiplexador pode ser encarado como um descodificador binário se:
 - A entrada de dados for considerada um enable adicional
 - As entradas de controlo forem consideradas as entradas binárias do descodificador

- Função
 - Codificar as palavras do código
- Síntese
 - Entradas: nº de palavras do código
 - Saídas: comprimento do código
- Características
 - Em cada instante, apenas **uma** entrada deve estar **ativa**
 - Como não é possível garantir esta restrição, é necessário atribuir **prioridades** às entradas

Codificador com prioridade

- Prioridade
 - Tipicamente as entradas com **maior peso** têm prioridade
- Saídas
 - Para poder distinguir da situação em que está activa a entrada menos prioritária, existe uma saída adicional que indica se alguma entrada está activa

Codificador com prioridade 4x2

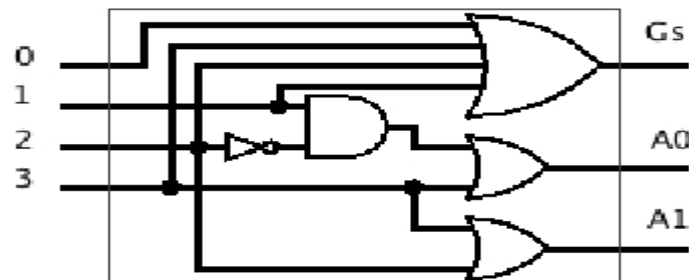
- Tabela de verdade

3	2	1	0	A1	A0	Gs
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

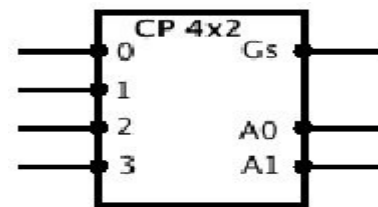
- Expressão algébrica

- $G_s = 0 + 1 + 2 + 3$
- $A_0 = 1 \bar{2} + 3$
- $A_1 = 2 + 3$

- Logigrama



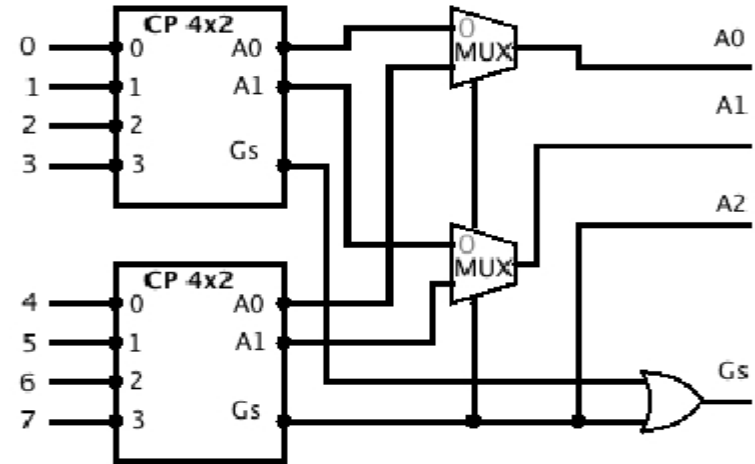
- Símbolo lógico



Expansão de codificador com prioridade

- Como construir?
 - Utilizar multiplexadores
- Exemplo
 - Construir um codificador com prioridade de 8 entradas a partir de codificadores de 4 entradas

- Circuito



Tarefas até à próxima aula prática

- Ficha 06 – Circuitos combinatórios
 - 3); 4); 5); 6); 7)