conversão

decimal → binário:

$x \div 2 = y \; R \; 1$ ↑

$y \div 2 = z \; R \; 0$

⋮

$0,x \cdot 2 = 1 + 0,y$ ↓

$0,y \cdot 2 = 0 + 0,z$

binário ← decimal:

1 0 0 1 , 1 1

$1.2^3 + 0.2^2 + 0.2^1 + 1.2^0 + 1.2^{-1} + 1.2^{-2}$

binário → hex:

agrupar de 4 em 4

$\underbrace{100}_{4}\underbrace{1111}_{F}$

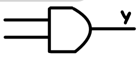
decimal → BCD:

cada algarismo vira 4 bits

binário ← hex:

agrupar de 4 em 4 bits

AND



Y = A . B

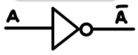
Y só é 1 quando 1 . 1

OR

Y = A + B

Y só é 0 quando 0 + 0

NOT

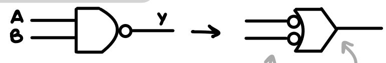


De Morgan

$\overline{A.B} = \bar{A} + \bar{B}$

$\overline{A+B} = \bar{A} . \bar{B}$

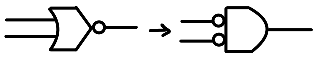
NAND



$Y = (\overline{A.B}) = \bar{A} + \bar{B}$ representação alternativa

Y só é 0 quando 1 . 1

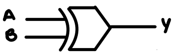
NOR



Y só é 1 quando $\overline{0+0}$

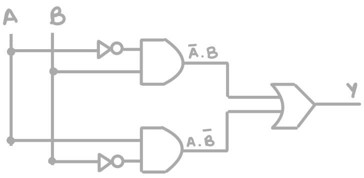
$Y = (\overline{A+B}) = \bar{A} . \bar{B}$

XOR (ou exclusivo)

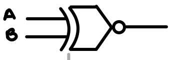


$Y = \bar{A}.B + A.\bar{B}$ $Y = A \oplus B$

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



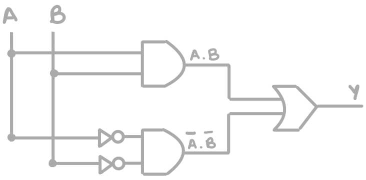
XNOR



$Y = A \odot B = (\bar{A}.\bar{B}) + (A.B)$

$= \overline{A \oplus B}$

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



álgebra booleana

A.B = B.A   A+B = B+A

A.(B+C) = A.B + A.C

A + (B.C) = (A+B).(A+C)

(A.B).C = A.(B.C)

(A+B)+C = A+(B+C) $\bar{\bar{A}} = A$

1. A = A   0+A = A

$A.\bar{A} = 0$ $A + \bar{A} = 1$ $AB = AB(C+\bar{C})$

A . A = A   A + A = A $= ABC + AB\bar{C}$

0. A = 0   1+A = 1

A.(A+B) = A $A.(\bar{A}+B) = A.B$

A+(A.B) = A $A + (\bar{A}.B) = A+B$

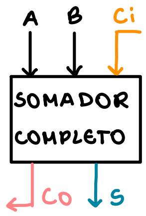
mapas de Karnaugh

	$\bar{B}$ 0	B 1
$\bar{A}$ 0	0	1
A 1	2	3

	$\bar{C}$ 0	C 1
$\bar{A}\bar{B}$ 00	0	1
$\bar{A}B$ 01	2	3
AB 11	6	7
$A\bar{B}$ 10	4	5

	$\bar{C}\bar{D}$ 00	$\bar{C}D$ 01	CD 11	$C\bar{D}$ 10
$\bar{A}\bar{B}$ 00	0	1	3	2
$\bar{A}B$ 01	4	5	7	6
AB 11	12	13	15	14
$A\bar{B}$ 10	8	9	11	10

Eliminam-se as variáveis que mudam de estado dentro do bloco ($x \to \bar{x}$ ou $\bar{x} \to x$).



S = fsop (Ci, A, B)

= {1, 2, 4, 7}

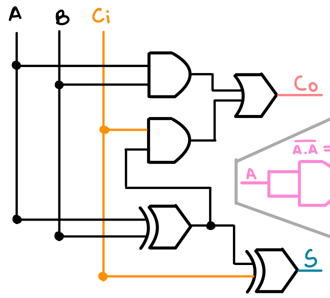
$S = Ci \oplus (A \oplus B)$

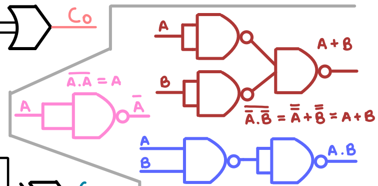
Co = fsop (Ci, A, B)

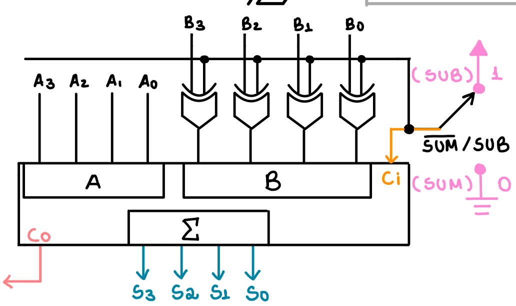
= {3, 5, 6, 7}

Co = A.B + Ci.Y

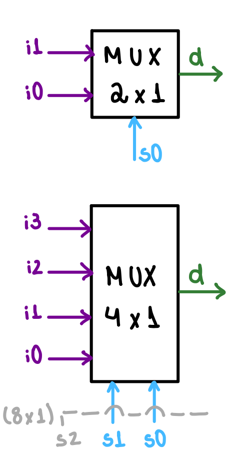
$Y = Ci(\bar{A}B + A\bar{B})$

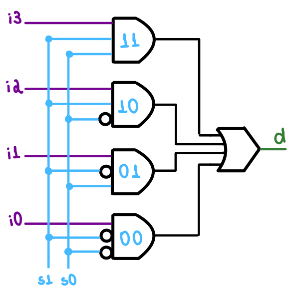






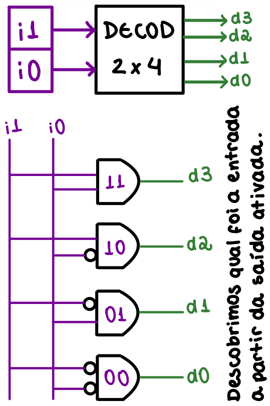
MUX → N entradas de dados e 1 saída





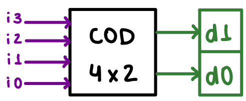
Nos seletores colocamos em binário a posição da entrada desejada.
Ex.: s1 = 1 e s2 = 0 seleciona a entrada i2 para passar.

DECODIFICADOR



Descobrimos qual foi a entrada a partir da saída ativada.

CODIFICADOR



Sempre existirá uma única entrada ativa por vez.
Se i2 = 1, então a saída será 10 (2 em binário).

d1 = i3 + i2

d0 = i3 + i1

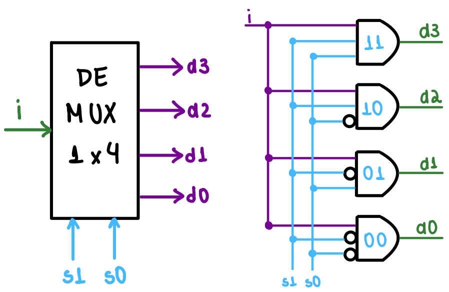
No caso do Enable Output, a prioridade é dada à entrada mais elevada que tem um 1.

i3	i2	i1	i0	d1	d0	e
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

e = i3 + i2 + i1 + i0

Ex.: se as entradas i3 e i2 forem iguais (de modo que as entradas são 1010), a prioridade é dada a i3 e a saída produzida é 11.

DEMUX → 1 entrada de dados e N saídas



Passa a entrada para a saída determinada pelos seletores. As outras saídas permanecem em 0.