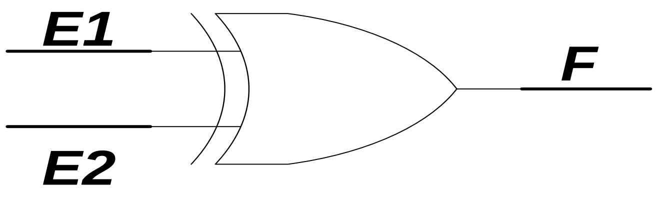
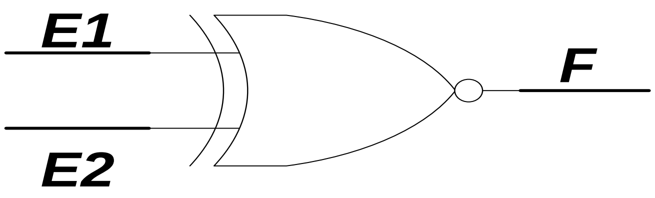
Propriedades da Porta XOR

Tabelas Verdade: XOR e XNOR





E1	E2	F
0	0	0
0	1	1
1	0	1
1	1	0

E1	E2	F
0	0	1
0	1	0
1	0	0
1	1	1

$$F = \overline{E1}\,E2 + E1\,\overline{E2}$$

$$F = \overline{E1}\;\overline{E2} + E1\,E2$$

Equações soma de produto

Propriedades da Porta XOR

a) $E1 \oplus (E2 \oplus E3) = (E1 \oplus E2) \oplus E3 = E1 \oplus E2 \oplus E3$

b) $E1 \oplus 1 = \overline{E1}$

c) $E1 \oplus 0 = E1$

d) $E1 \oplus E1 = 0$

e) $E1 \oplus \overline{E2} = \overline{E1 \oplus E2} = E1 \odot E2$

Prova

$$\overline{E1 \oplus E2} = \overline{\overline{E1}\,E2 + E1\,\overline{E2}} = (E1 + \overline{E2})\,(\overline{E1} + E2) =$$

$$\underbrace{E1\,\overline{E1}}_{0} + E1\,E2 + \overline{E2}\,\overline{E1} + \underbrace{\overline{E2}\,E2}_{0} = E1\,E2 + \overline{E2}\;\overline{E1} = E1 \oplus \overline{E2}$$

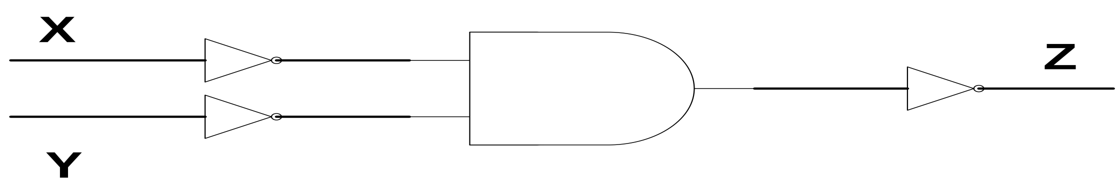
OPERADOR UNIVERSAL

Suficiência das operações:

As operações básicas: "E", "OU" e "NÃO"

a) **Operação OU → operações (E,NÃO)**

$$Z = X + Y = \overline{\overline{X + Y}} = \overline{\overline{X}\ \overline{Y}}$$



Portanto, as operações "E" e "NÃO" são suficientes para representar qualquer função lógica

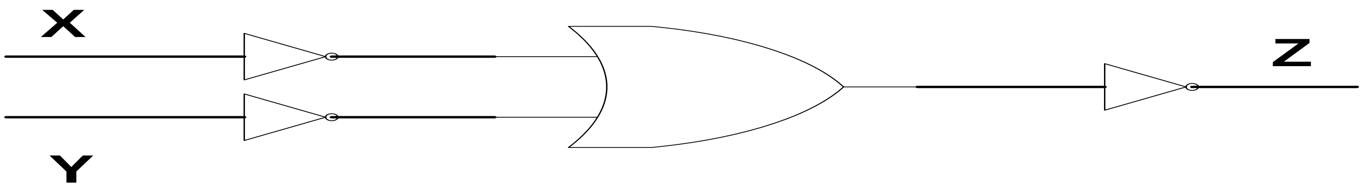
OPERADOR UNIVERSAL

Suficiência das operações:

As operações básicas: "E", "OU" e "NÃO"

b) **Operação E → operações (OU,NÃO)**

$$Z = X\,Y = \overline{\overline{X\,Y}} = \overline{\overline{X} + \overline{Y}}$$



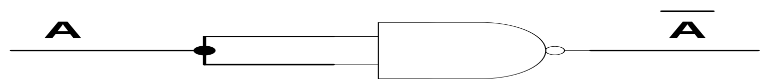
Portanto, as operações "OU" e "NÃO" são suficientes para representar qualquer função lógica

OPERADOR UNIVERSAL

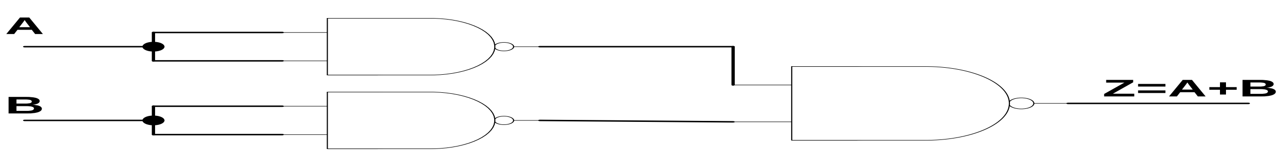
Suficiência das operações:

c) **Operações (OU,NÃO) → operação NAND**

i) $$Z=\overline{A\ A}=\overline{A}$$



ii) $$Z=\overline{\overline{A\ A}\ \overline{B\ B}}=\overline{\overline{AA}}+\overline{\overline{BB}}=AA+BB=A+B$$



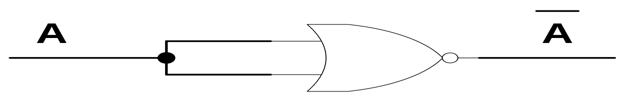
Portanto, a operação NAND é suficiente para representar qualquer função lógica → *Porta Universal*

OPERADOR UNIVERSAL

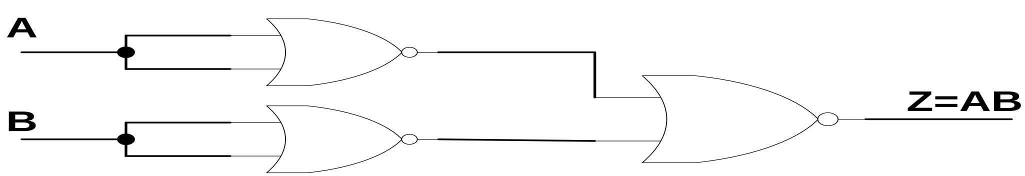
Suficiência das operações:

d) **Operações (E,NÃO) → operação NOR**

I) $Z=\overline{A+A}=\overline{A}$



II) $Z=\overline{\overline{A+A}+\overline{B+B}}=\overline{\overline{A+A}}\;\overline{\overline{B+B}}=A+A\;\;B+B=A\;B$

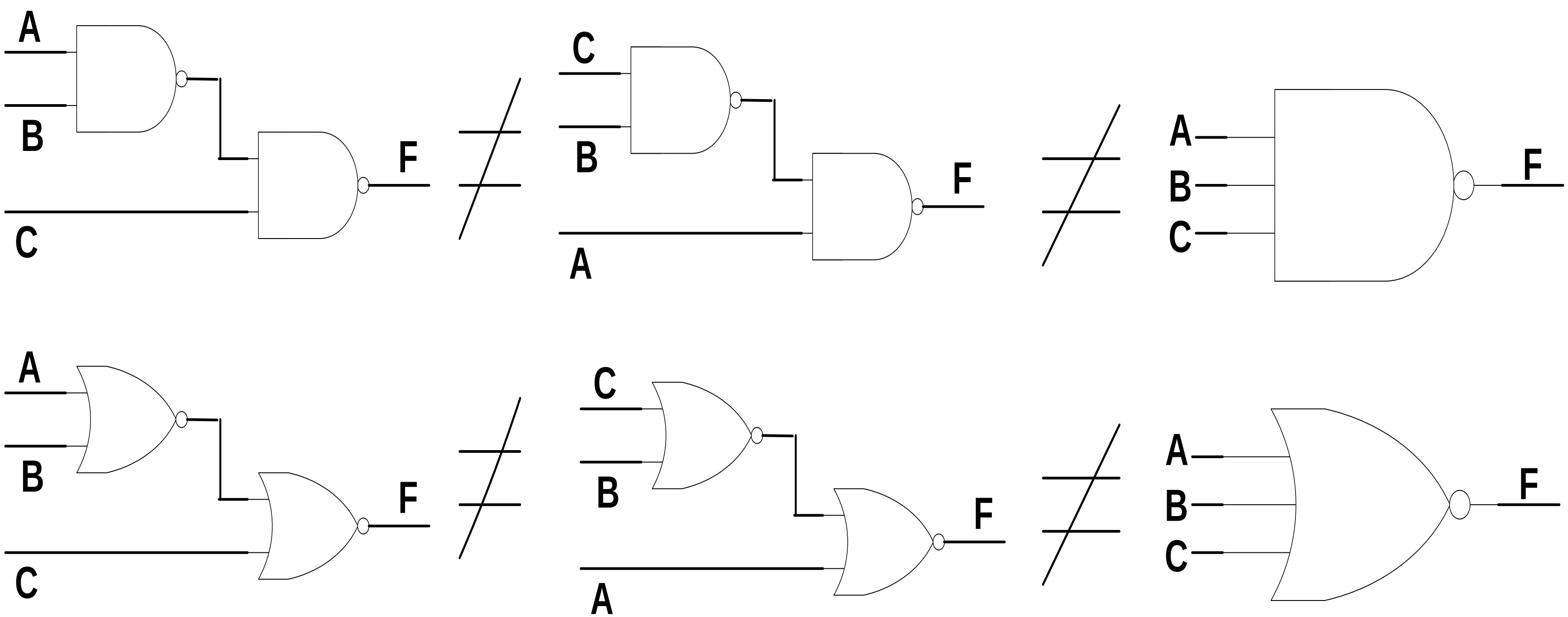


Portanto, a operação NOR é suficiente para representar qualquer função lógica → *Porta Universal*

OPERADOR UNIVERSAL

Suficiência das operações:

Operações (NAND, NOR → não satisfazem a propriedade associativa



Conversão: SOP →POS

Exemplo: F=AB + BC'

1) Mapa de Karnaugh → usar maxtermos

F=ABC + ABC' + ABC' + A'BC'

C \ AB	00	01	11	10
0	0	1	1	0
1	0	0	1	0

F= B ▪ (A + C')

Conversão: SOP→POS

2) **Teorema de DeMorgan**

$F = AB + BC'$

$F' = (AB + BC')'$

$F' = (AB)' \bullet (BC')'$

$F' = (A' + B') \bullet (B' + C)$

$F' = A'B' + A'C + B' + B'C$

$F' = B' + A'C$

$F = (B' + A'C)'$

$F = B \bullet (A'C)'$

$F = B \bullet (A + C')$

Conversão: SOP → POS

3) **Principio da Dualidade**

$E = X + 0$ → dual trocar 0 por 1 e trocar + por • (vice-versa)

$D[E] = X \bullet 1$

$F = AB + BC' \Rightarrow D[F] = (A + B) \bullet (B + C')$

$D[F] = AB + AC' + B + BC'$

$D[F] = B + AC'$

$D[D[F]] = B \bullet (A + C')$

Procedimento: uso da porta XOR

Exemplo:

$$F=(A,B,C,D)=\sum(3,4,8,14) + d_m(2,5,9,15)$$

Implementar usando somente portas XOR

(usando mapa de karnaugh)

CD \ AB	00	01	11	10
00	0	1	0	1
01	0	x	0	x
11	1	0	x	0
10	x	0	1	0

II

I

II) $A'B\,C'D' + A'BC'D + A'B'C\,D + A'B'C\,D'$

$A'BC'(D + D') + \quad A'B'C\,(D + D')$

$A'(B\,C' + B'C) = A'(B \oplus C)$

I) $AB'C'D' + AB'C'D + ABCD + ABCD'$

$AB'C'(D + D') \quad + \quad ABC\,(D + D')$

$A\,(B'C' + BC) = A(B \oplus C)'$

$F=A'X + AX' = A \oplus X = A \oplus B \oplus C$

Funções Combinatórias de Múltiplas Saídas

Projetar um comparador de 1 bit

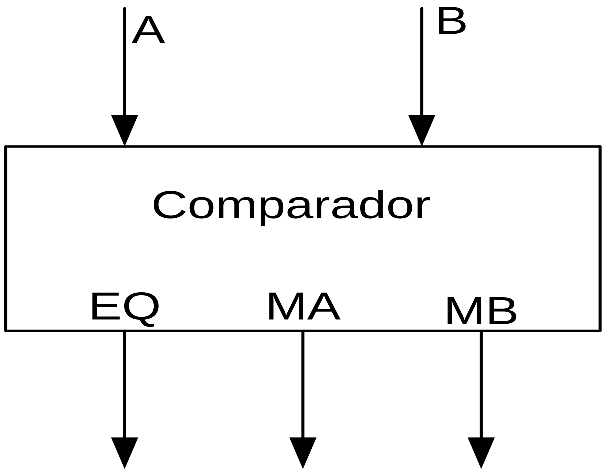


Tabela Verdade

A B	EQ	MA	MB
0 0	1	0	0
01	0	0	1
11	1	0	0
10	0	1	0

Equações Booleanas

EQ= A'B'+ AB = (A⊕B)'

MA = AB'

MB = A'B

Minimização: Funções de Múltiplas Saídas

Exemplo: $F1(A,B,C)=\sum(0,2,3,5,6)$

$F2(A,B,C)=\sum(1,2,3,4,7)$

$F3(A,B,C)=\sum(2,3,4,5,6)$

1) Minimização individual

C \ AB	00	01	11	10
0	1	1	1	0
1	0	1	0	1

C \ AB	00	01	11	10
0	0	1	0	1
1	1	1	1	0

C \ AB	00	01	11	10
0	0	1	1	1
1	0	1	0	1

F1=A'C' + A'B + AB'C + BC'

F2=A'B + A'C + BC + AB'C'

F3=A'B + BC' + AB'

Número de Literais= 18 e Número de termos=8

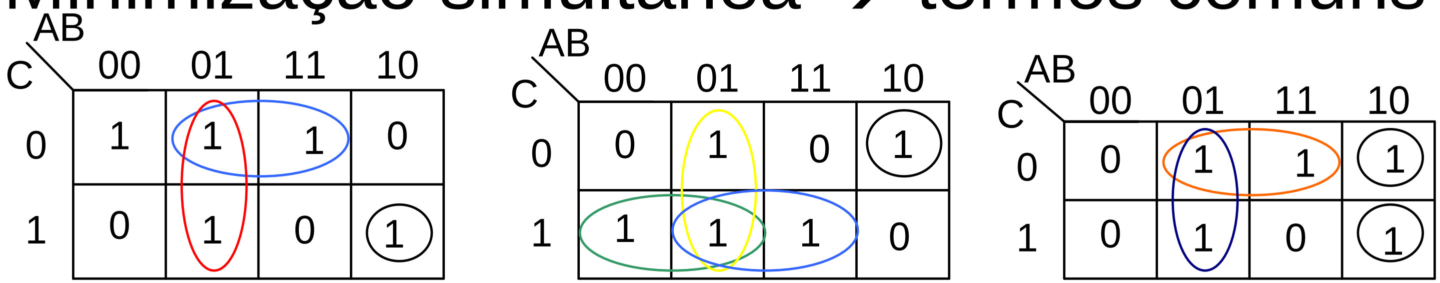
Minimização: Funções de Múltiplas Saídas

Exemplo: F1(A,B,C)=∑(0,2,3,5,6)

F2(A,B,C)=∑(1,2,3,4,7)

F3(A,B,C)=∑(2,3,4,5,6)

2) Minimização simultânea → termos comuns



F1=A’C’ + A’B + AB’C + BC’ F2=A’B + A’C + BC + AB’C’ F3=A’B + BC’ + AB’C’ + AB’C

Número de Literais= 16 e Número de termos= 7

Funções Combinatórias: Especiais

1) Codificadores: código decimal → binário

Exemplo 4x2

E_0	E_1	E_2	E_3	Y_1	Y_0
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

Tabela Verdade

$$Y_1=E_0'E_1'E_2E_3' + E_0'E_1'E_2'E_3$$

$$Y_0=E_0'E_1E_2'E_3'+ E_0'E_1'E_2'E_3$$

Equações Booleanas SOP

Funções Combinatórias: Especiais

2) Codificadores de prioridade

Exemplo 4x2

E_0	E_1	E_2	E_3	Y_1	Y_0
1	0	0	0	0	0
x	1	0	0	0	1
x	x	1	0	1	0
x	x	x	1	1	1

Tabela Verdade

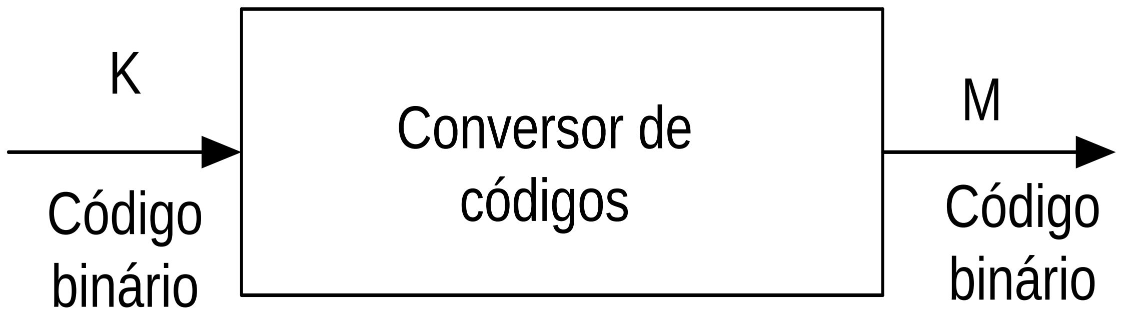
$$Y_1 = E_3 + E_2E_3' = E_3 + E_2$$

$$Y_0 = E_3 + E_1E_2'E_3' = E_3 + E_1E_2'$$

Equações Booleanas SOP

Funções Combinatórias: Especiais

3) Conversor de códigos

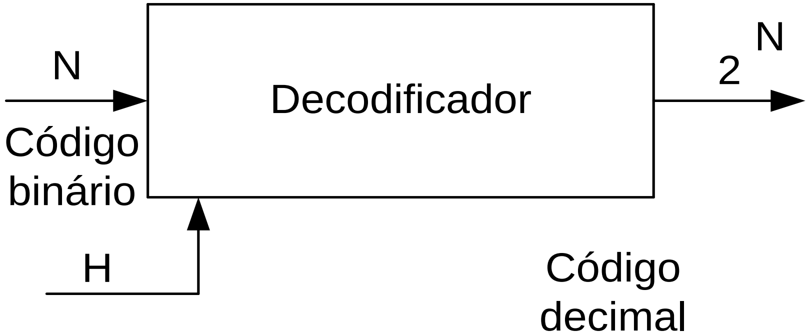


Exemplos: Códigos binário puro → código de Gray
Código binário puro → código one-hot
Codígo binário puro → códfigo BCD (binary codified decimal)

Códigos alfa-númericos: ASCII, etc.

Funções combinatórias: Especiais

4) Decodificadores: código binário → decimal



Exemplo 2 x 4

H	E_1	E_2	$\overline{Y_0}$	$\overline{Y_1}$	$\overline{Y_2}$	$\overline{Y_3}$
1	0	0	0	1	1	1
1	0	1	1	0	1	1
1	1	0	1	1	0	1
1	1	1	1	1	1	0
0	x	x	0	0	0	0

Tabela Verdade

$Y0=H(E_1 + E_2)$ $Y1=H(E_1 + E_2')$

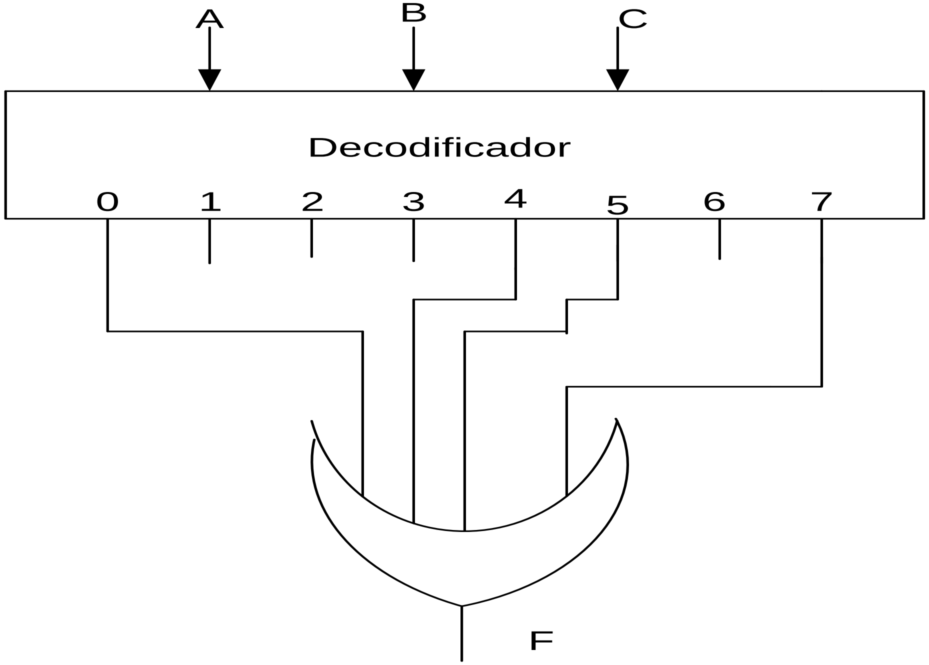
$Y2=H(E_1' + E_2)$ $Y3=H(E_1'+ E_2')$

Equações Booleanas POS

Funções combinatórias: Especiais

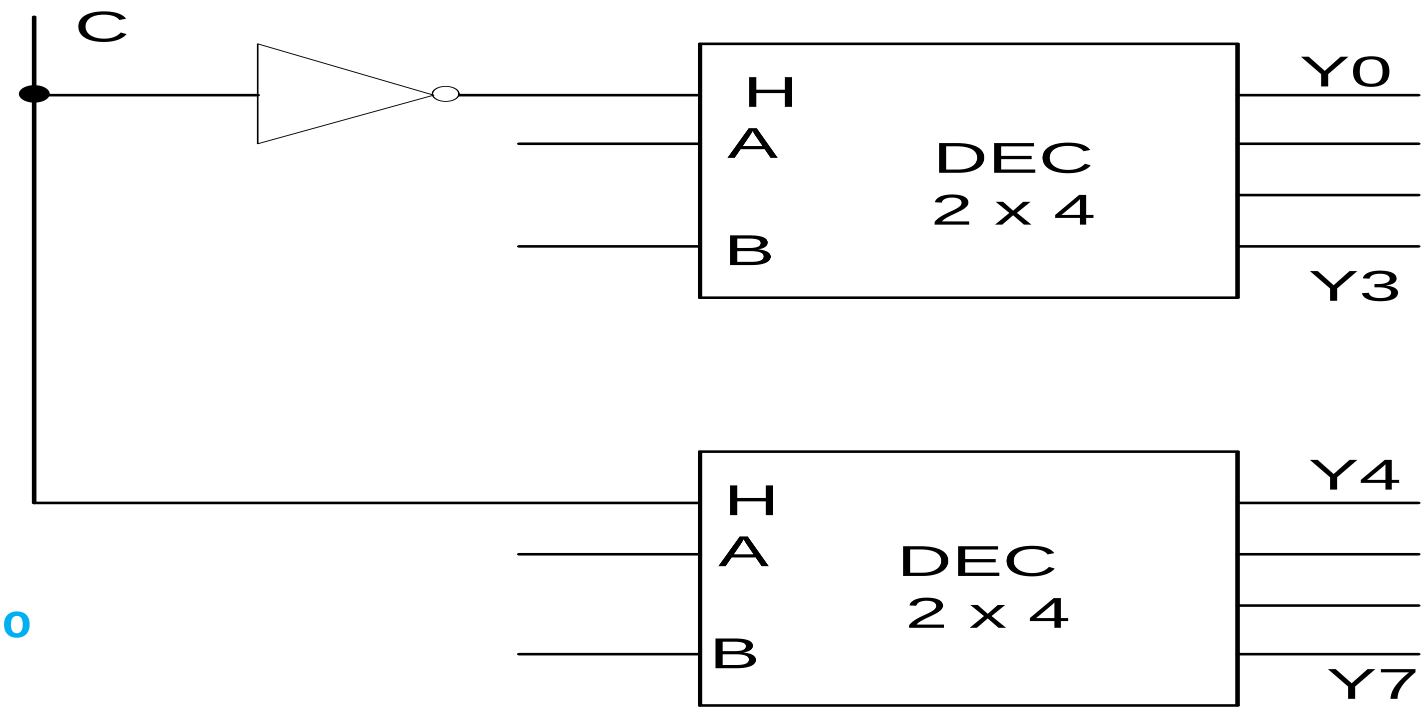
Decodificadores→ **Aplicação**: **Gerador de função**

Exemplo: F(A,B,C)=∑(0,4,5,7)



Funções Combinatórias: Especiais

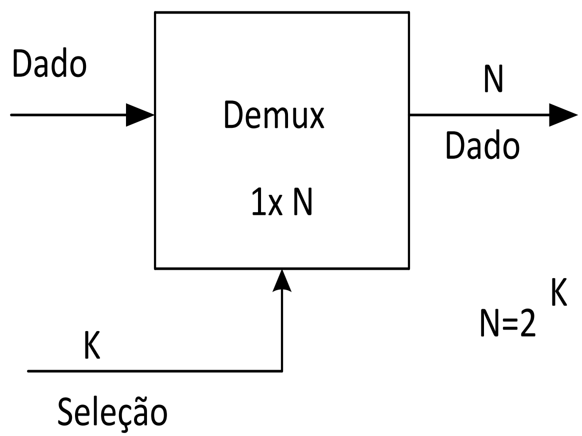
Expansão de decodificadores: 3x8 usando 2x4



Onde C é o mais significativo

Funções Combinatórias: Especiais

5) **Demultiplexador**: comunicação de dados

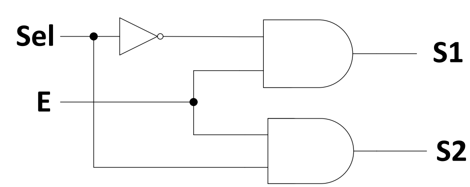


Exemplo: 1x2

Sel	E	S1	S2
0	0	0	0
0	1	1	0
1	0	0	0
1	1	0	1

$S1 = Sel' \, E$

$S2 = Sel \, E$



Funções Combinatórias: Especiais

Expansão de Demux: 1x4 usando 1x2