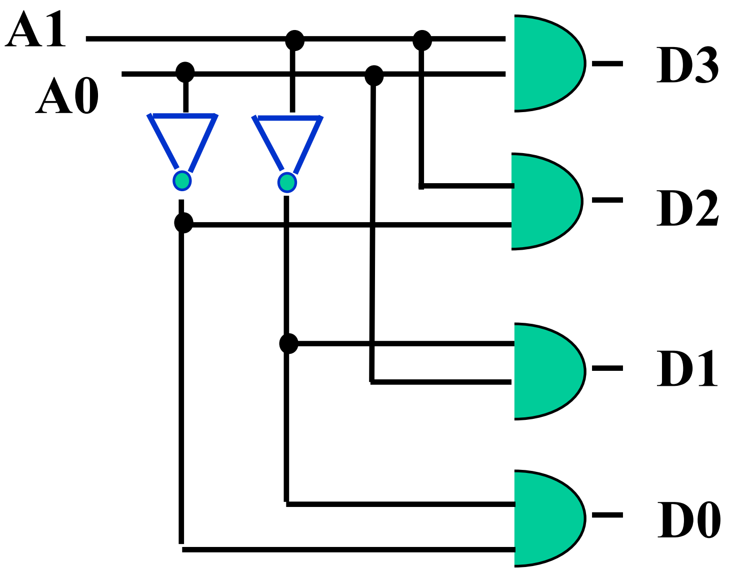
INF01118

Técnicas Digitais para Computação

Decodificador/Codificador
Multiplexador

Aula 16

Decodificadores



A_1	A_0	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Imaginando-se que A_1A_0 seja um código, as saídas o decodificam.

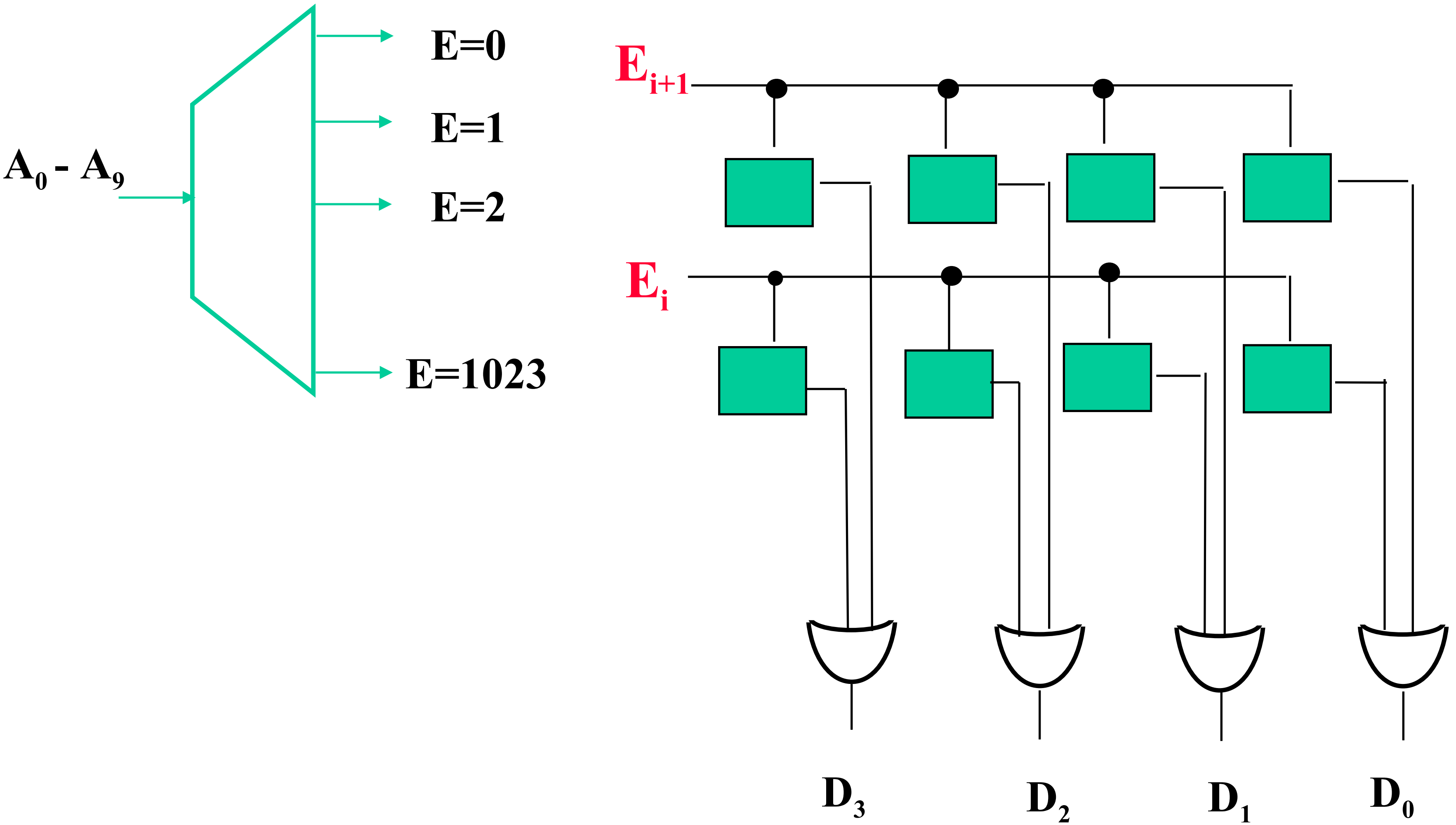
Se cada saída tivesse uma lâmpada (LED) que acendesse quando esta saída fosse igual a 1, e se esta lâmpada iluminasse um número com o algarismo decodificado, teríamos explicitamente na saída a informação sobre o código detectado.

Apenas uma saída é igual a 1
Ex: se $A_1A_0 = 10$ então $D_2 = 1$

"2"

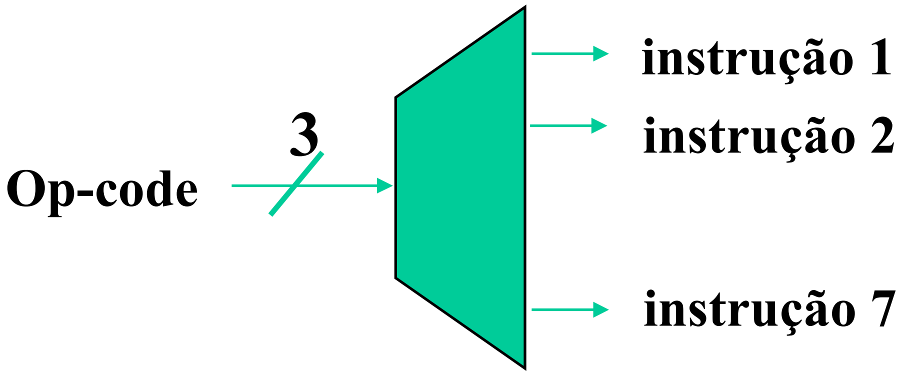
Aplicações

- **Decodificação de endereço em uma memória**



- **Decodificador de instruções**

Formato de uma instrução

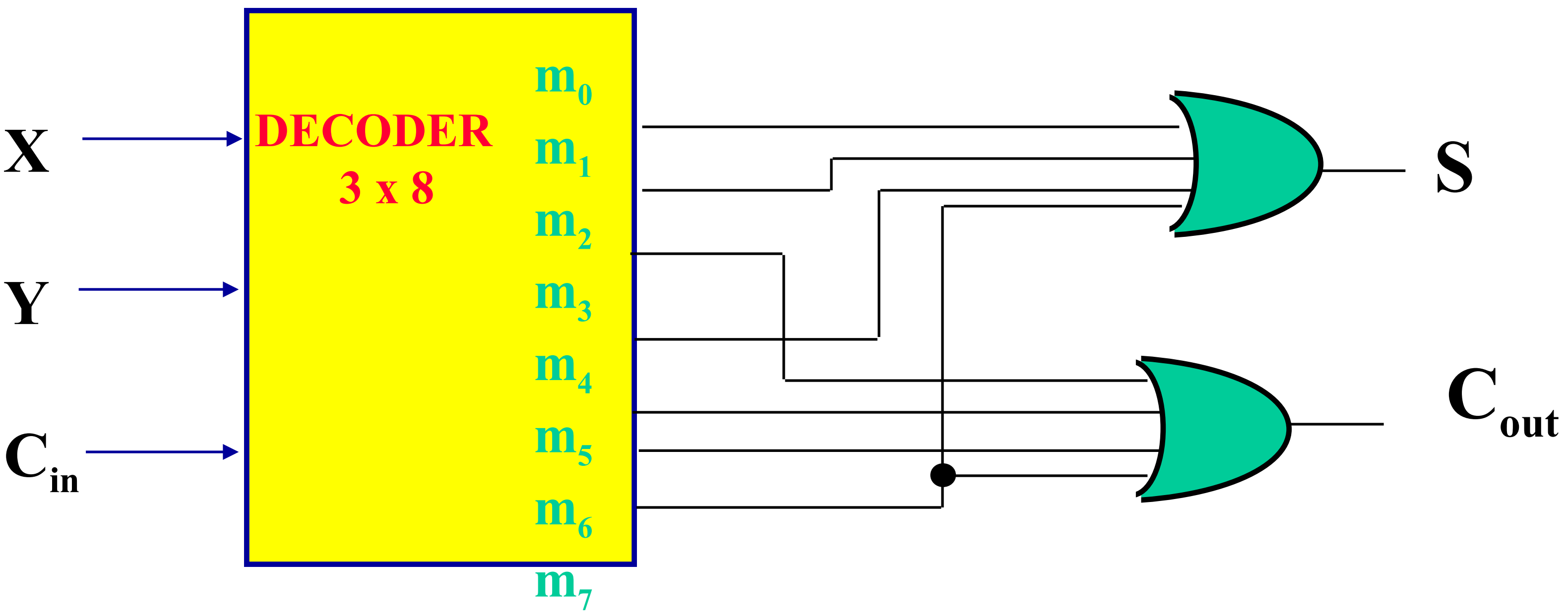


- **Implementação de funções combinacionais**
  - Imaginando que as entradas são as variáveis X,Y,Z de uma função F
  - As 8 saídas correspondem então aos mintermos
  - Tomando a soma dos produtos usando mintermos, basta fazer um OR entre os mintermos para os quais a função é = 1.

Exemplo: Full Adder

$S(X,Y,C_{in}) = \Sigma\, m\ (1,2,4,7)$

$C_{out}\ (X,Y,C_{in}) = \Sigma\, m\ (3,5,6,7)$



Codificadores

Codificador simples

- 2^n entradas ➜ n saídas
- Apenas uma entrada pode ter valor = 1
- Saída fornece código binário correspondente à entrada ligada.

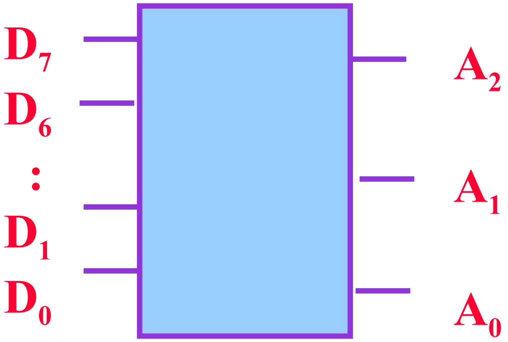


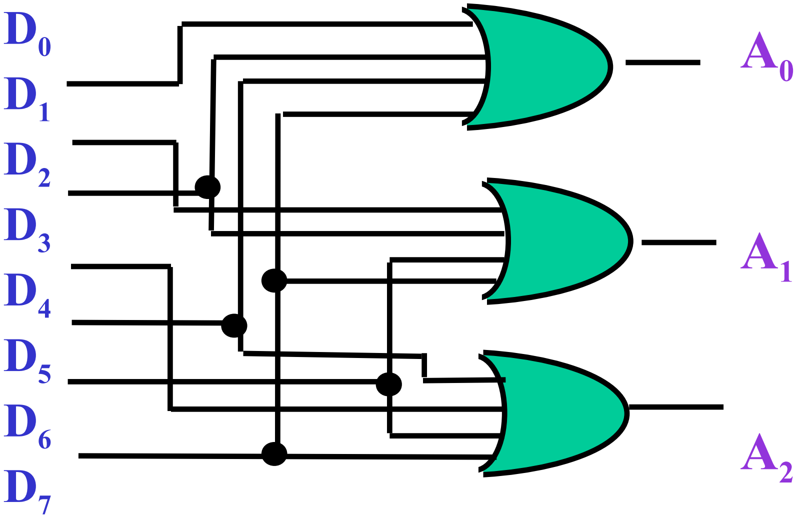
Tabela-verdade

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	A_2	A_1	A_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
:	:	:	:	:	:	:	:	:	:	:
1	0	0	0	0	0	0	0	1	1	1

Implementação

$$A_0 = D_1 + D_3 + D_5 + D_7$$
$$A_1 = D_2 + D_3 + D_6 + D_7$$
$$A_2 = D_4 + D_5 + D_6 + D_7$$

3 Portas OR
de 4 entradas



Problemas

Se mais de uma entrada = 1

Ex: $D_3 = 1$ e $D_6 = 1$

$A_2\ A_1\ A_0 = 1\ 1\ 1$ ⇨ como se $D_7 = 1$

Se nenhuma entrada = 1

$A_2A_1A_0 = 000$, como se D_0=1

Codificador de prioridade

Se duas entradas são iguais a 1 simultaneamente, a entrada de maior prioridade tem precedência.

Tabela - verdade

D_3	D_2	D_1	D_0	A_1	A_0	V
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

V indica saída válida
(pelo menos uma entrada = 1)

X = don't care

Mapas de Karnaugh

A_1

D_3D_2 \ D_1D_0	00	01	11	10
00				
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

A_0

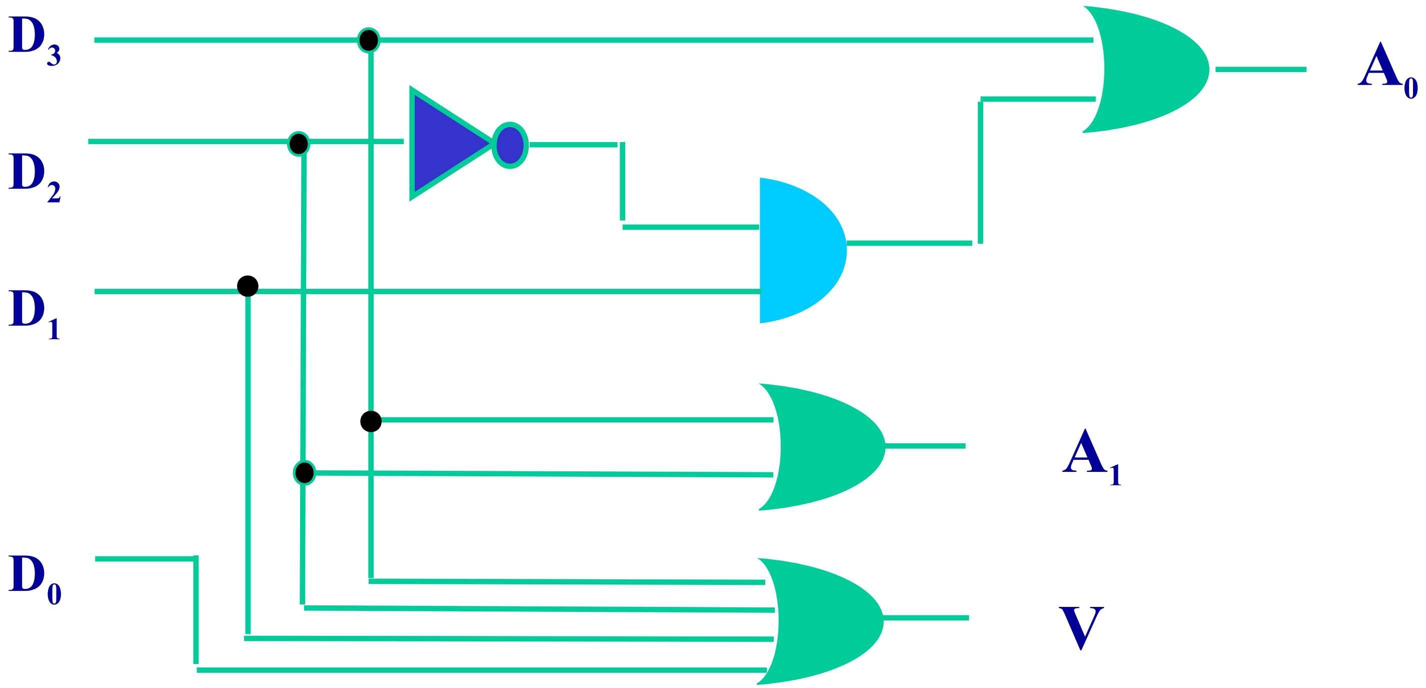
D_3D_2 \ D_1D_0	00	01	11	10
00			1	1
01				
11	1	1	1	1
10	1	1	1	1

$$A_1 = D_2 + D_3$$

$$V = D_3 + D_2 + D_1 + D_0$$

$$A_0 = D_3 + D_1 . \overline{D_2}$$

Implementação



Multiplexadores

- Seleciona 1 de 2^n entradas e a conecta à saída
- Seleção controlada por n sinais de controle

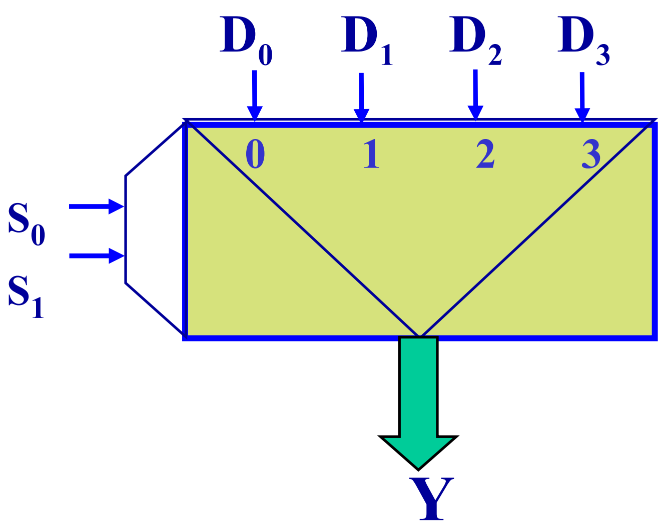
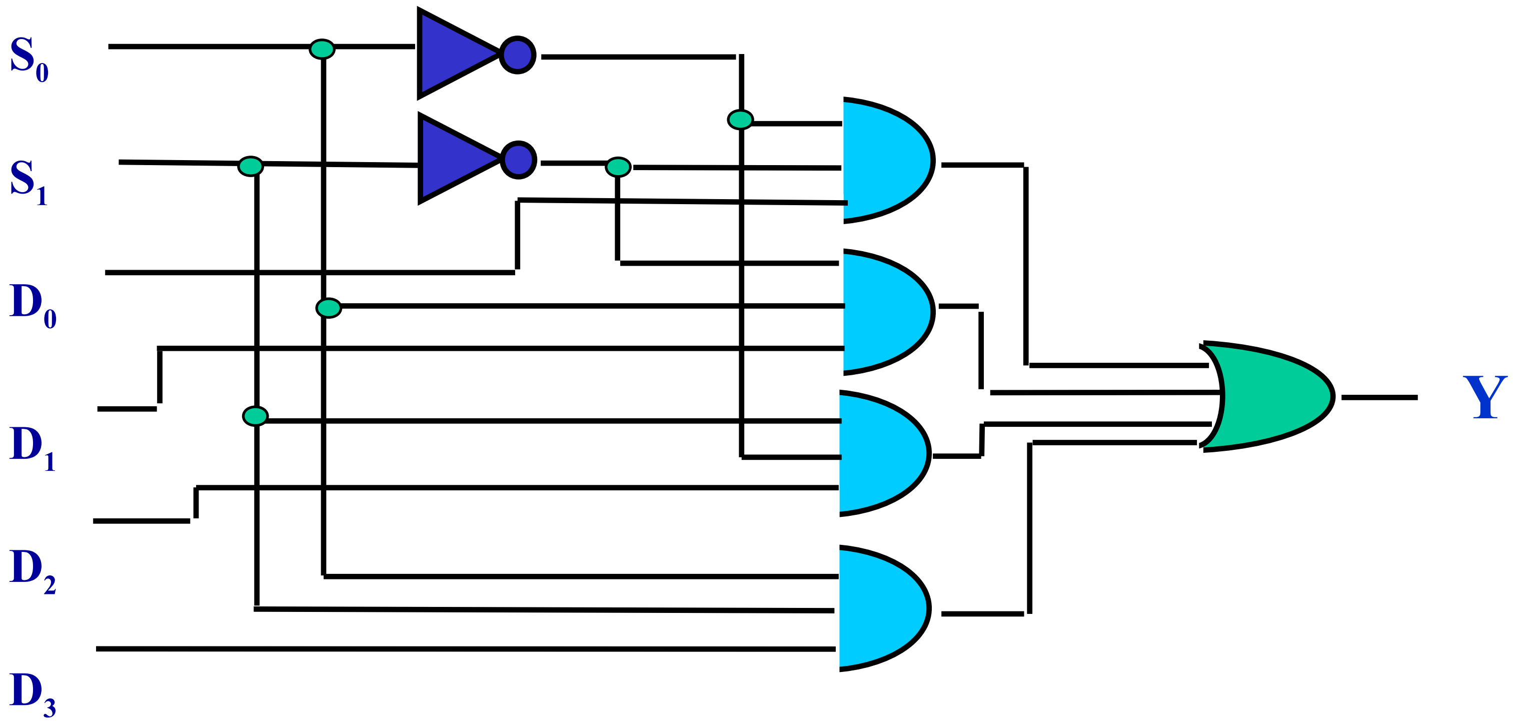


Tabela de Função

S_1	S_0	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

$$Y = \overline{S_0}\,\overline{S_1}D_0 + S_0\overline{S_1}D_1 + \overline{S_0}S_1D_2 + S_0S_1D_3$$

Implementação



Um mux é um decodificador ao qual foram acrescentados

- Uma entrada de dados D_0-D_3 em cada AND

  (portanto decodifica S_0 - S_1 e deixa passar o D_i correspondente)

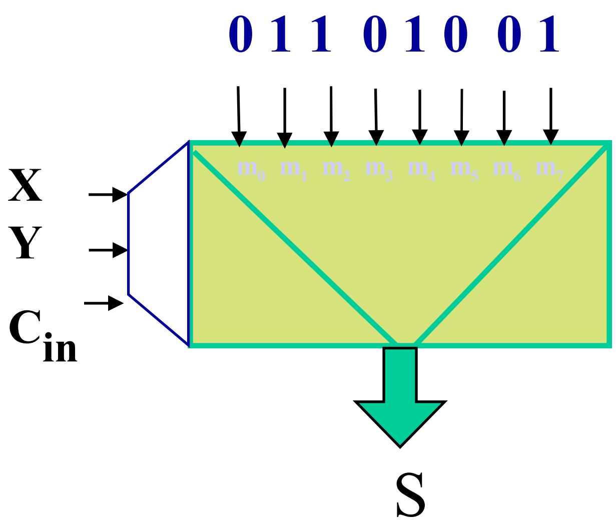
- Uma porta OR na saída

Implementação de funções booleanas

- **Alternativa 1**
  - Usando decoder: coloca-se na saída um OR dos mintermos desejados
  - Considerar que o MUX é um decoder que já tem o OR

  Portanto: usar MUX, selecionando mintermos através das entradas de dados

- Exemplo de somador: $S = \Sigma m\ (1,2,4,7)$



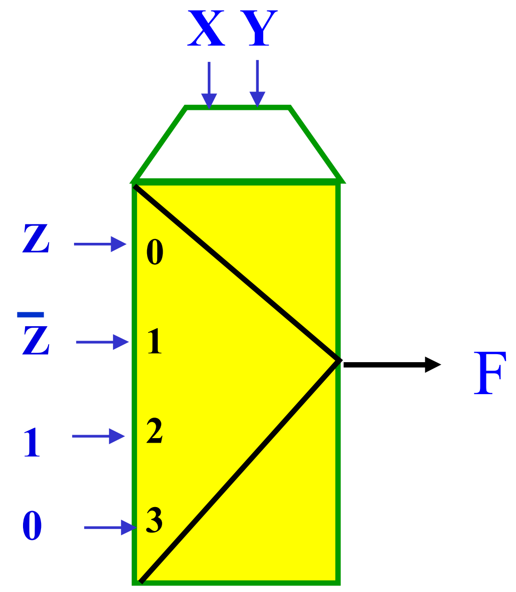
- Esta solução exige, para n variáveis, um MUX de n entradas de seleção.

• Alternativa 2

- Solução exige, para **n** variáveis , um MUX de **n-1** entradas de seleção (metade do tamanho de um MUX com **n** entradas de seleção)
- Método
  - Aplicar **n-1** primeiras variáveis como entradas de seleção
  - Usar variável restante (Z) como entrada de dados
  - Expressar saída como função de $\overline{Z}$, Z, 0, 1 (as únicas 4 alternativas que existem)
- Exemplo

  $F = \Sigma\, m(1,2,4,5)$

X	Y	Z	F	
0	0	0	0	F=Z
0	0	1	1	
0	1	0	1	$F=\overline{Z}$
0	1	1	0	
1	0	0	1	F=1
1	0	1	1	
1	1	0	0	F=0
1	1	1	0	



- **Exemplo do somador:** **$S = \Sigma\, m(1,2,4,7)$**

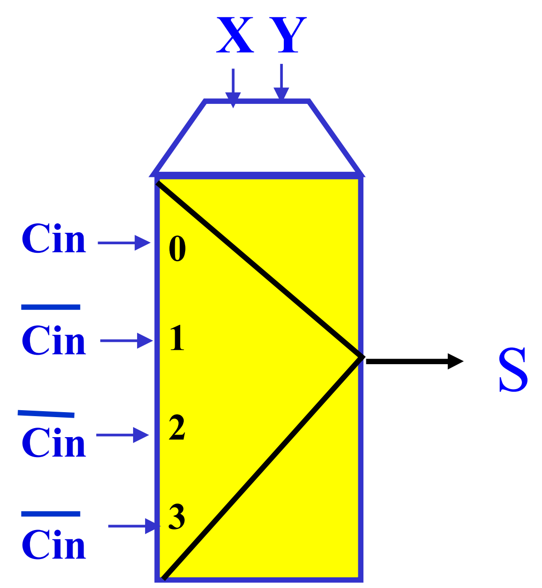
X	Y	Cin	S
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

$S=C_{in}$

$S=\overline{C}_{in}$

$S=\overline{C}_{in}$

$S=\overline{C}_{in}$



- **Outra aplicação de multiplexadores:**
  Seleção de caminhos de dados