Lecture 7

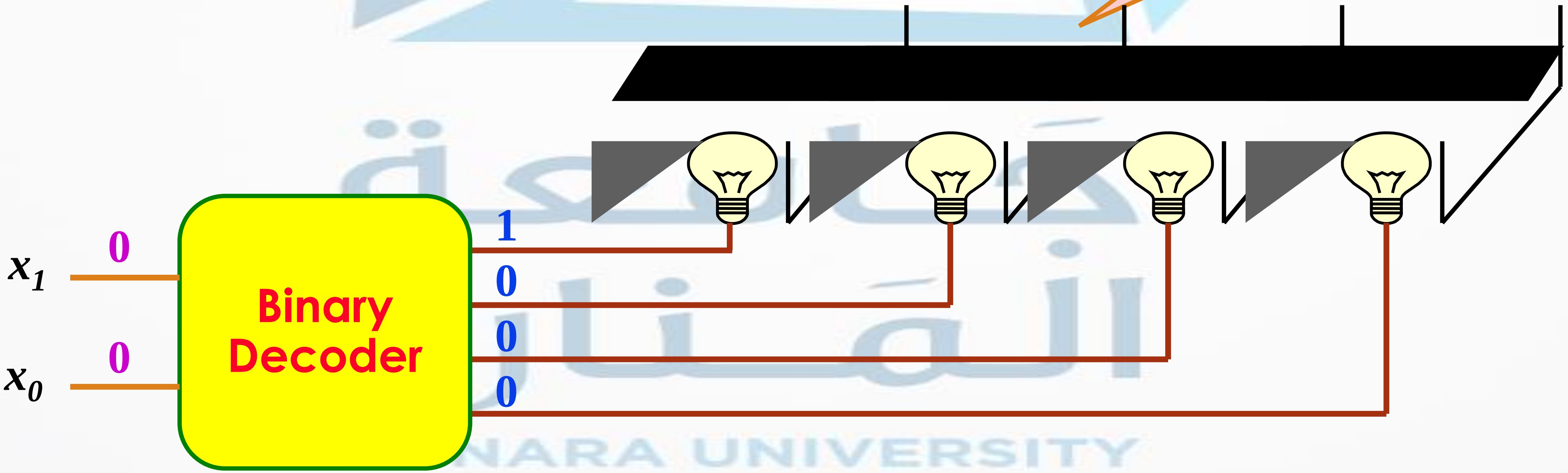
- Decoders
- Encoders
- Multiplexers
- DeMultiplexers
- Three-State Gates

Dr. Bassam Atieh

Decoders

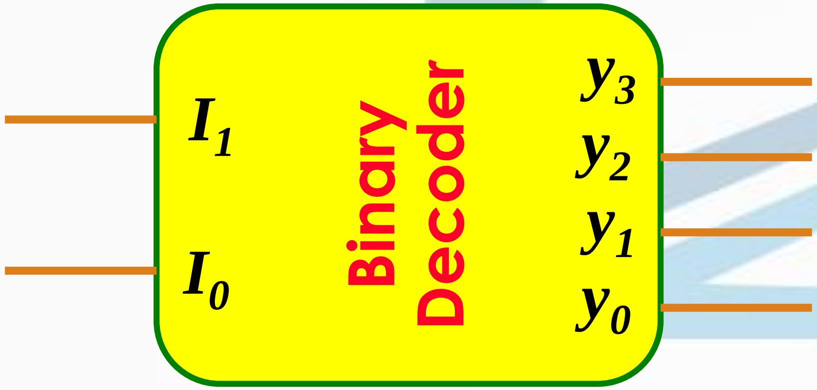
- Extract "*Information*" from the code
- Binary Decoder
  - Example: 2-bit Binary Number

Only *one* lamp will turn on

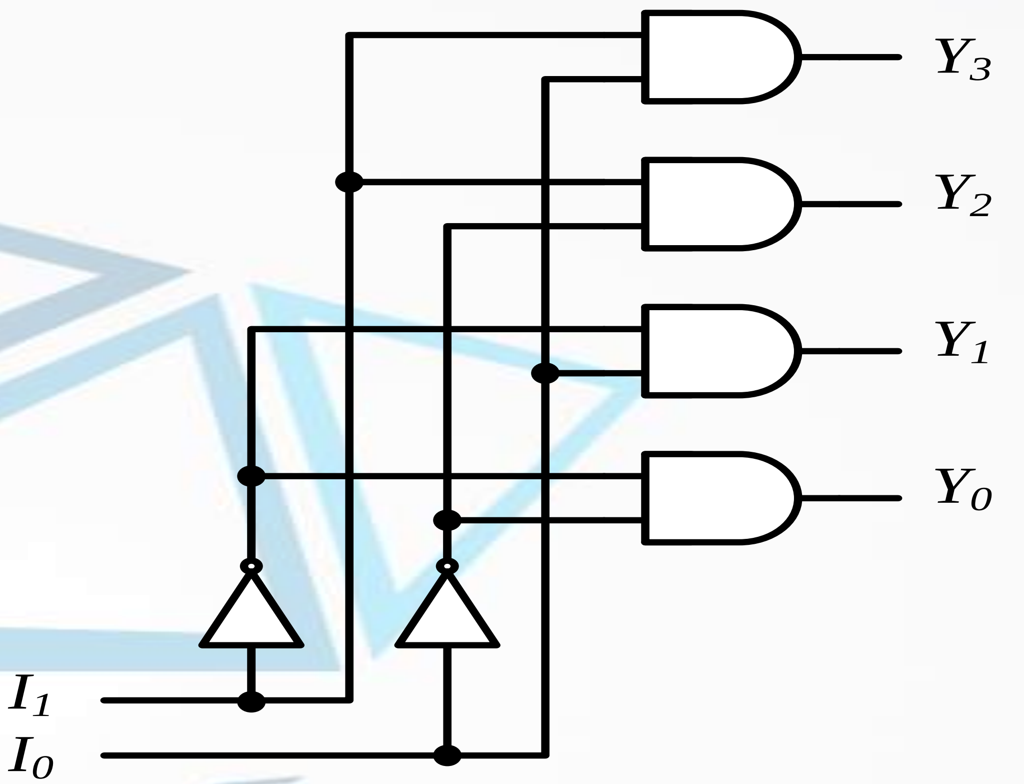


Decoders

- 2-to-4 Line Decoder



I_1 I_0	Y_3 Y_2 Y_1 Y_0
0 0	0 0 0 1
0 1	0 0 1 0
1 0	0 1 0 0
1 1	1 0 0 0

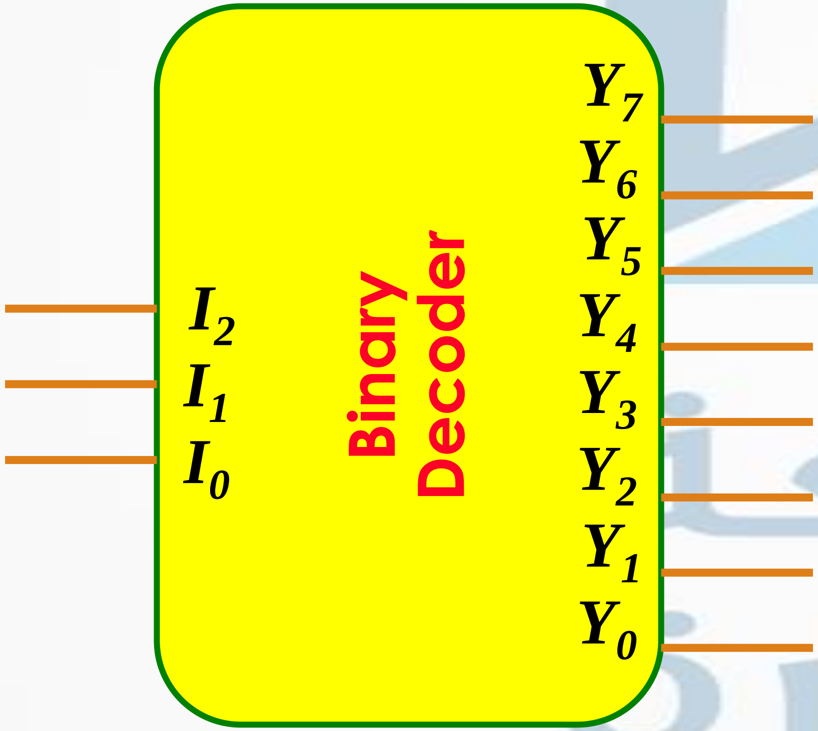


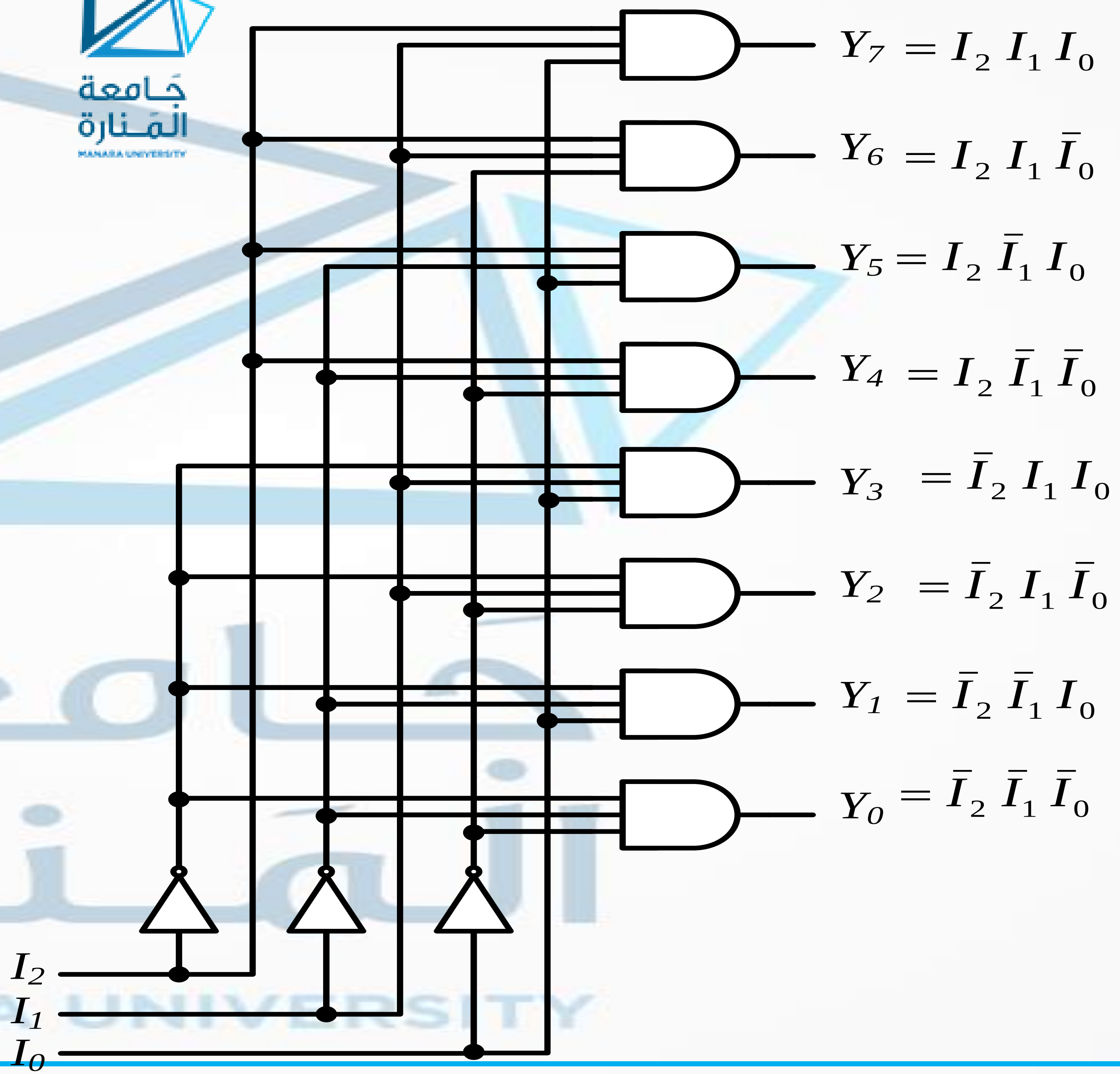
$$Y_3 = I_1 I_0 \qquad Y_2 = I_1 \bar{I}_0$$

$$Y_1 = \bar{I}_1 I_0 \qquad Y_0 = \bar{I}_1 \bar{I}_0$$

Decoders

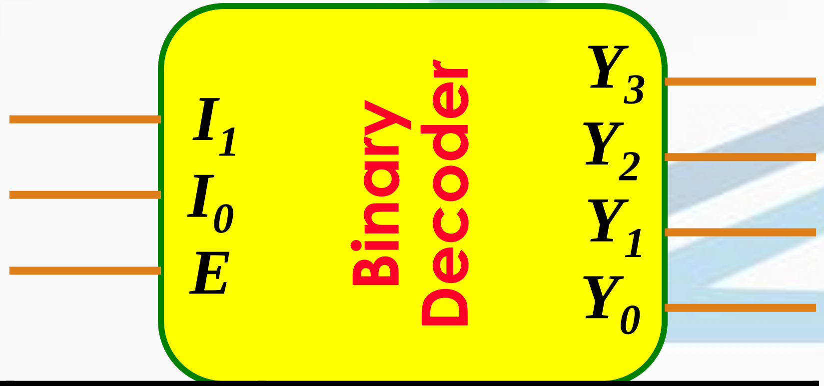
- 3-to-8 Line Decoder



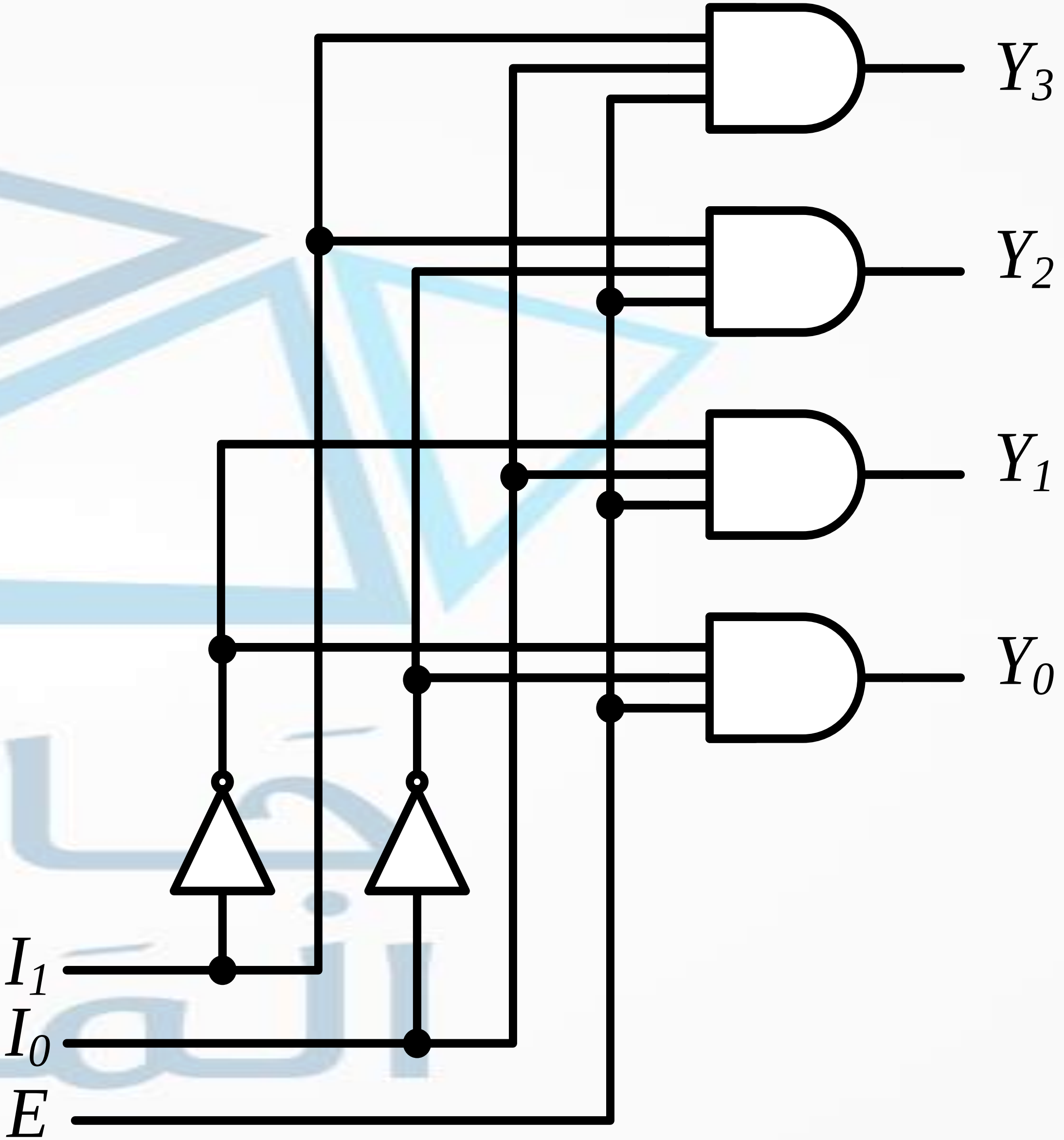


Decoders

- "*Enable*" Control



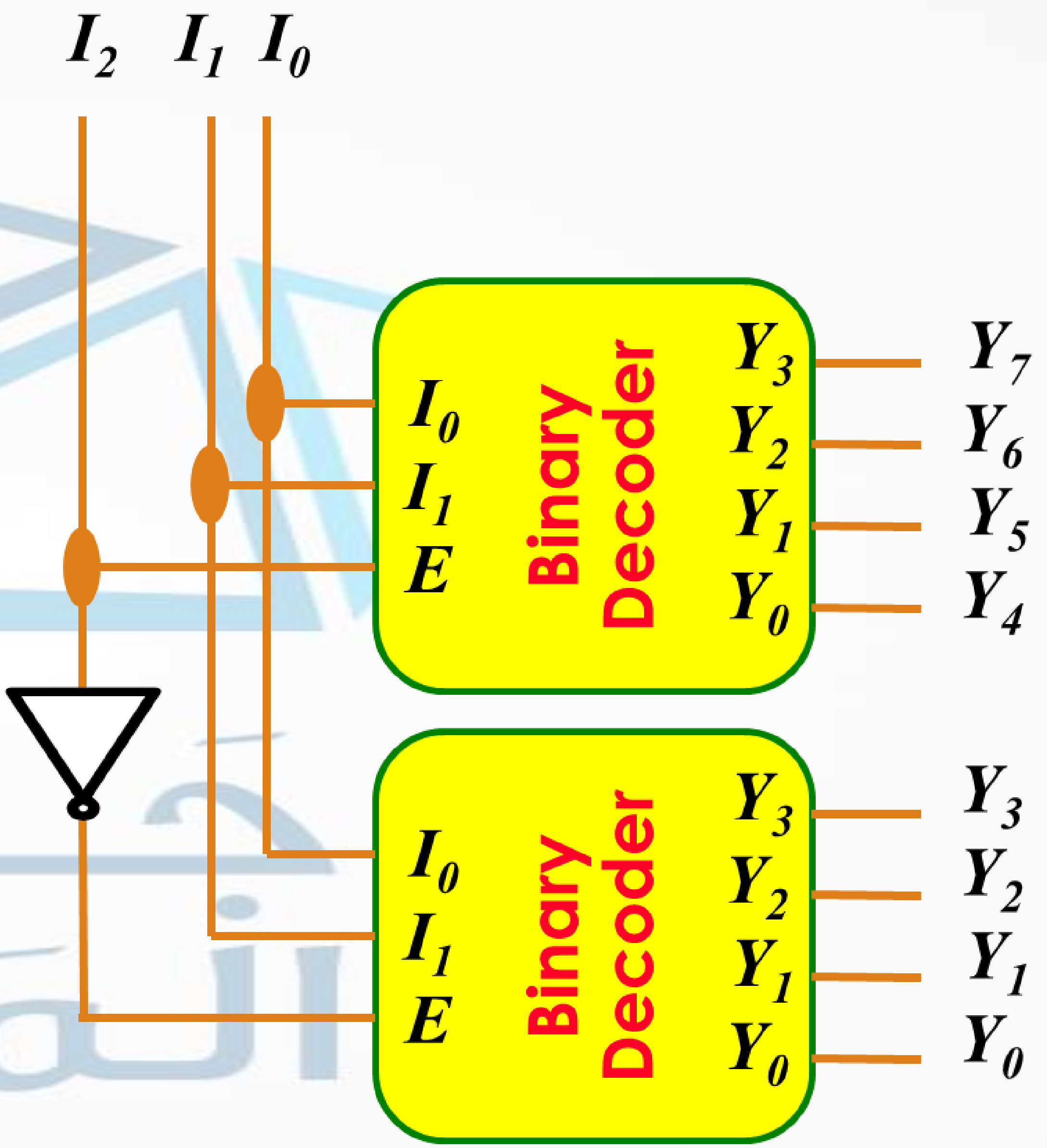
E	I_1	I_0	Y_3	Y_2	Y_1	Y_0
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0



Decoders

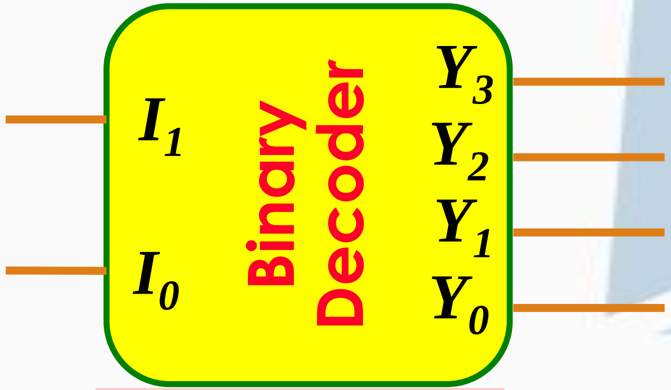
- Expansion

$I_2\ I_1\ I_0$	$Y_7\ Y_6\ Y_5\ Y_4\ Y_3\ Y_2\ Y_1\ Y_0$
0 0 0	0 0 0 0 0 0 0 1
0 0 1	0 0 0 0 0 0 1 0
0 1 0	0 0 0 0 0 1 0 0
0 1 1	0 0 0 0 1 0 0 0
1 0 0	0 0 0 1 0 0 0 0
1 0 1	0 0 1 0 0 0 0 0
1 1 0	0 1 0 0 0 0 0 0
1 1 1	1 0 0 0 0 0 0 0



Decoders

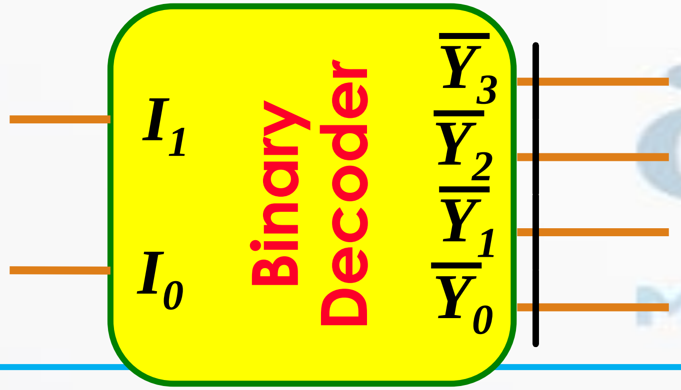
- Active-High / Active-Low



I_1 I_0	Y_3 Y_2 Y_1 Y_0
0 0	0 0 0 1
0 1	0 0 1 0
1 0	0 1 0 0
1 1	1 0 0 0

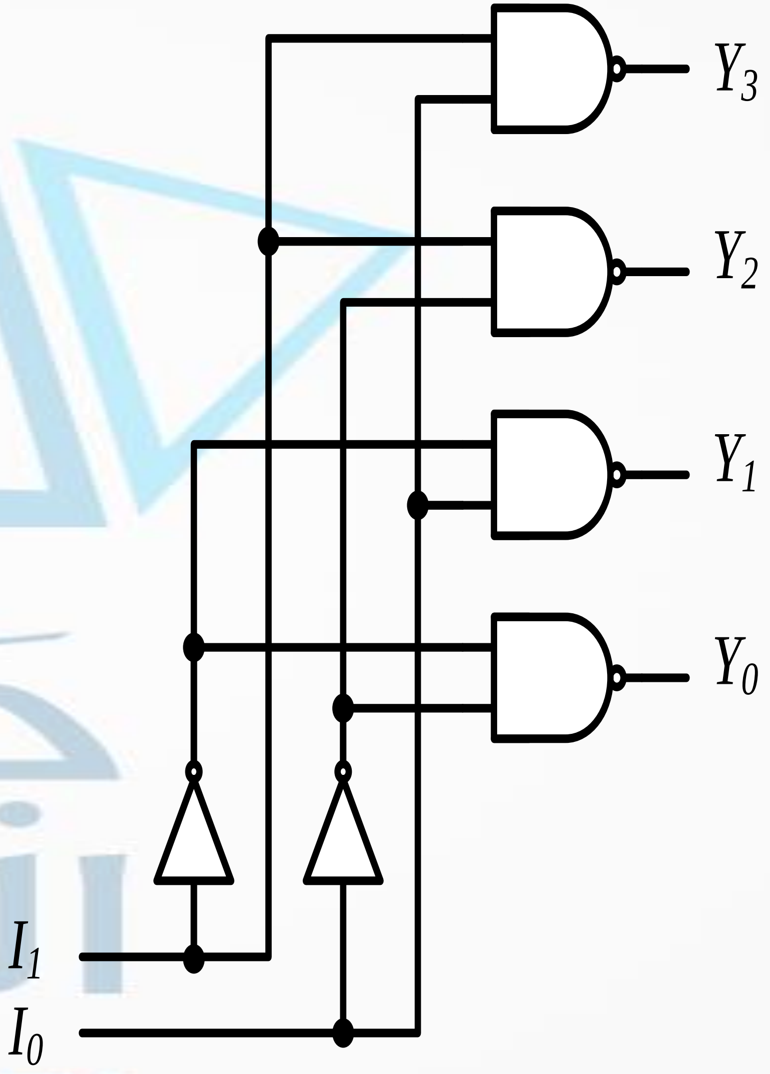
Active-High

Active-Low



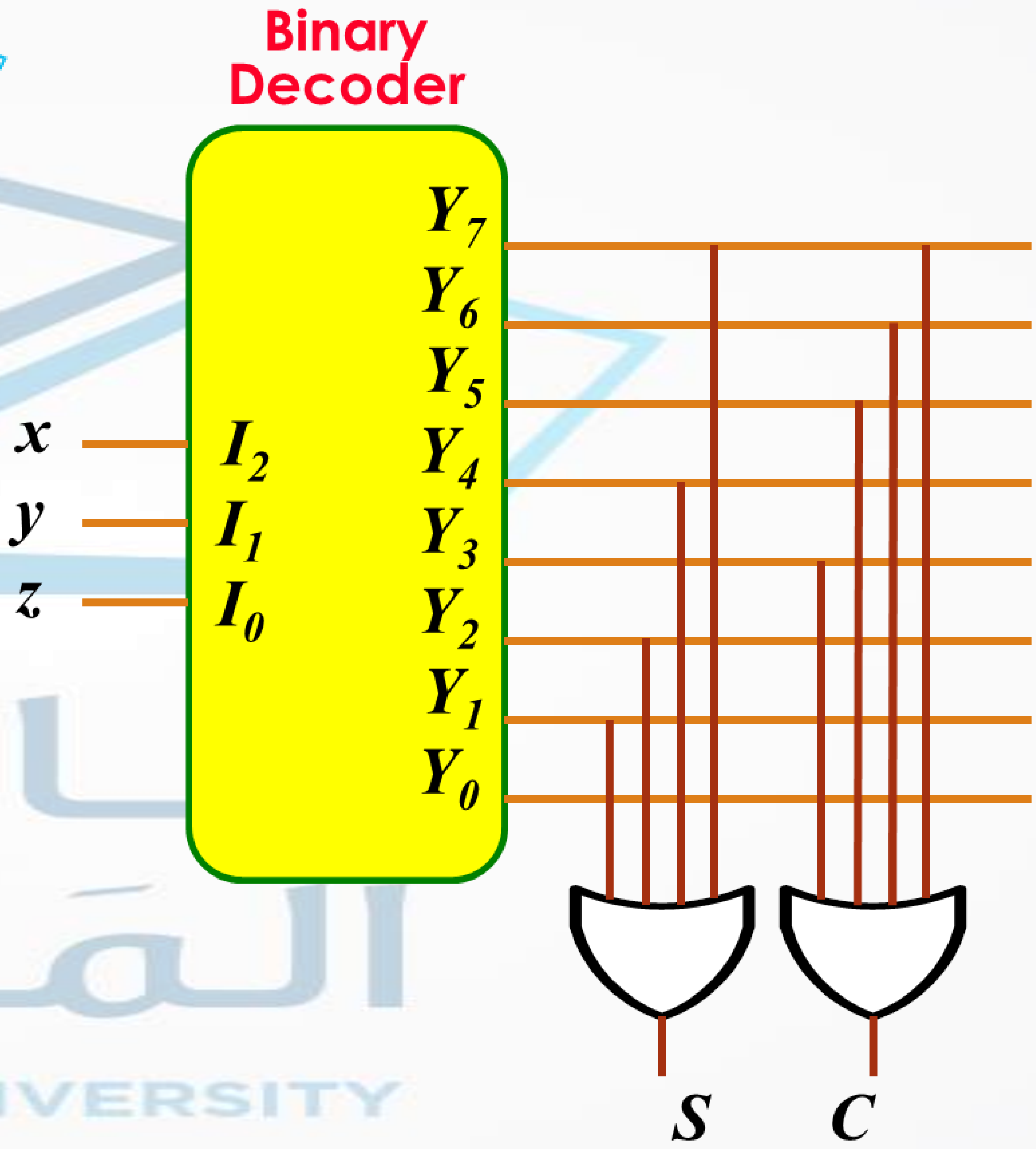
I_1 I_0	Y_3 Y_2 Y_1 Y_0
0 0	1 1 1 0
0 1	1 1 0 1
1 0	1 0 1 1
1 1	0 1 1 1

Active-Low



Implementation Using Decoders

- Each output is a minterm
- All minterms are produced
- Sum the required minterms

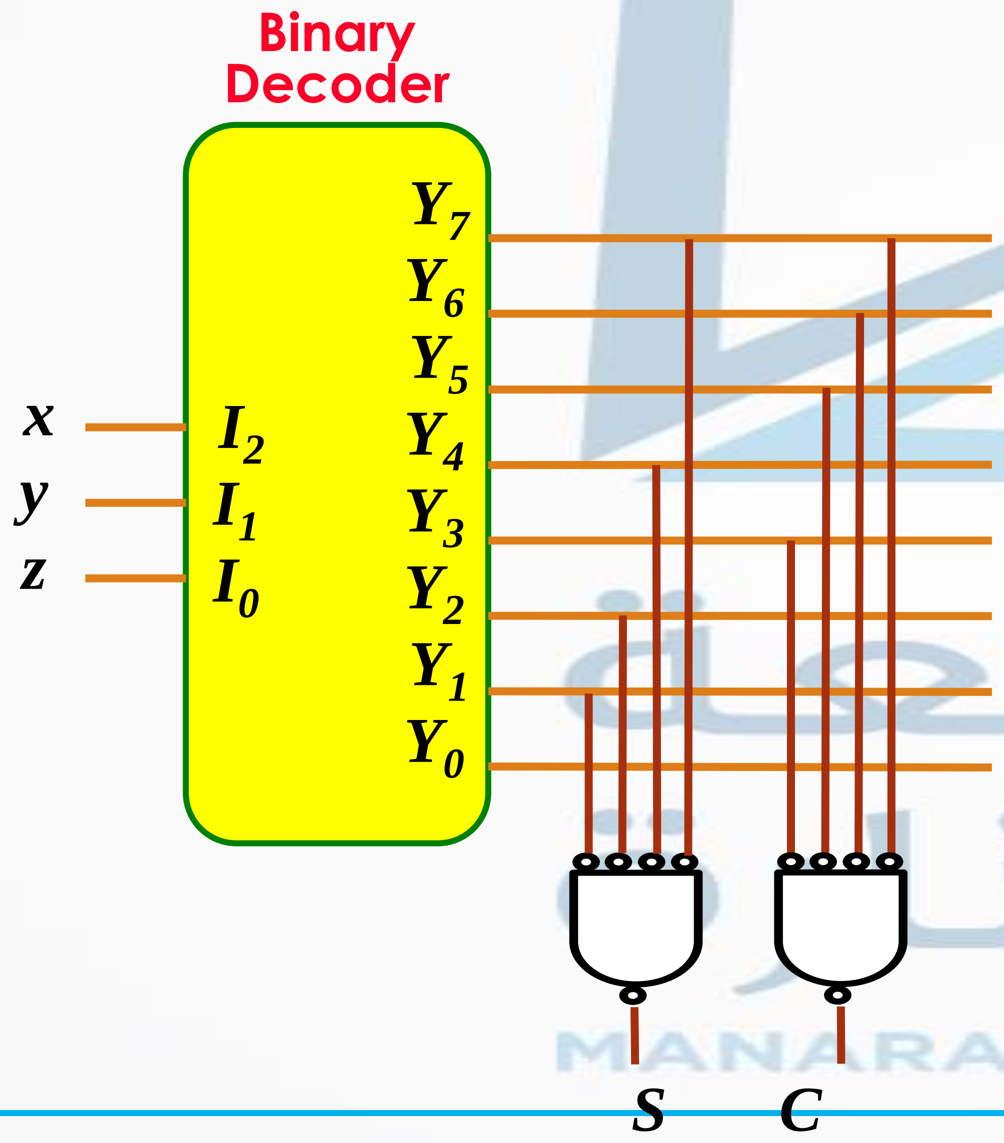


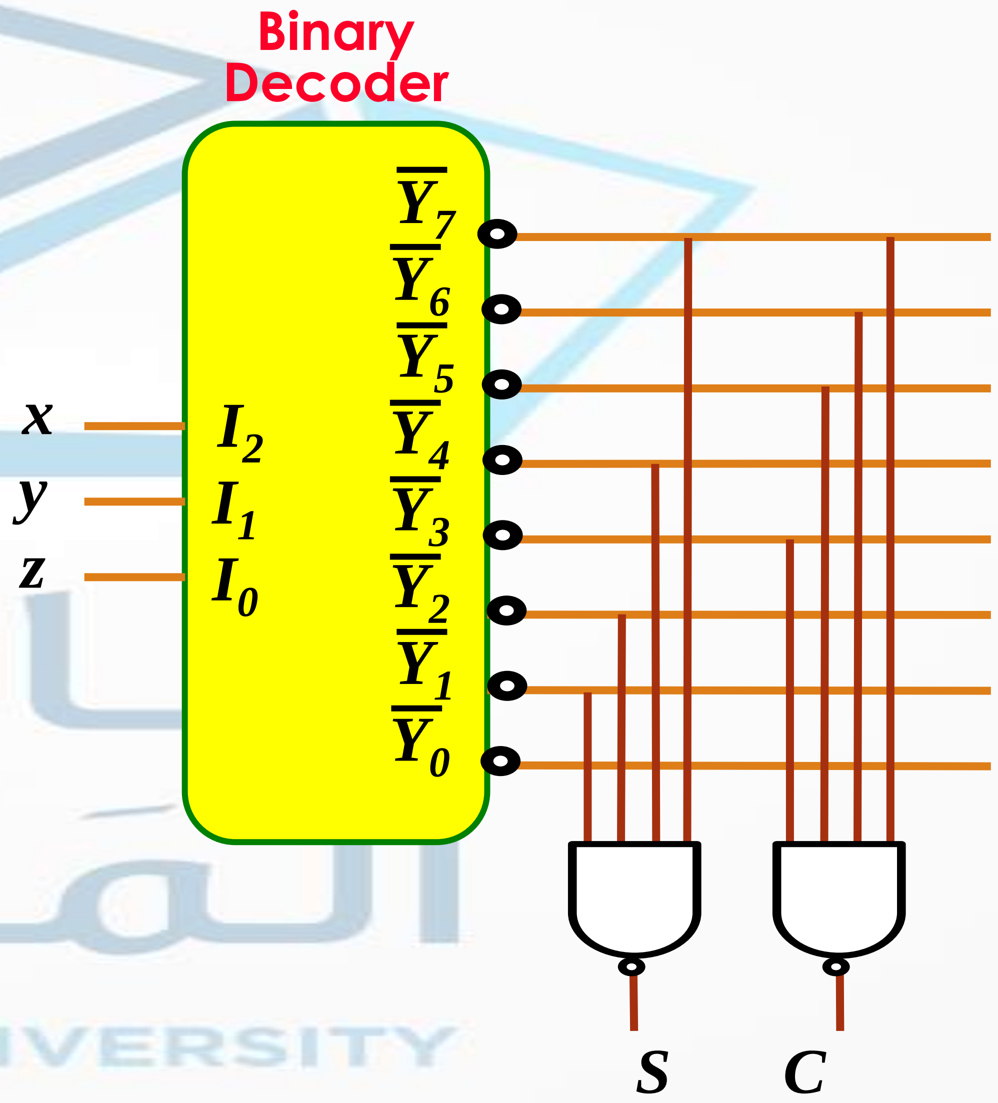
Example: Full Adder

$S(x, y, z) = \sum(1, 2, 4, 7)$

$C(x, y, z) = \sum(3, 5, 6, 7)$

Implementation Using Decoders

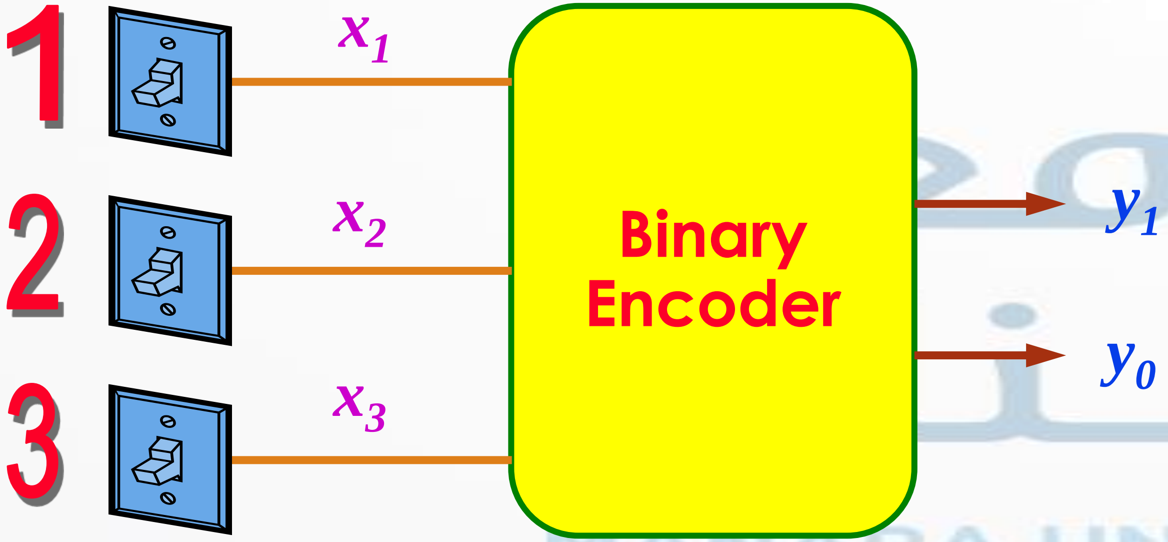




Encoders

- Put "*Information*" into code
- Binary Encoder
  - Example: 4-to-2 Binary Encoder

Only *one* switch should be activated at a time



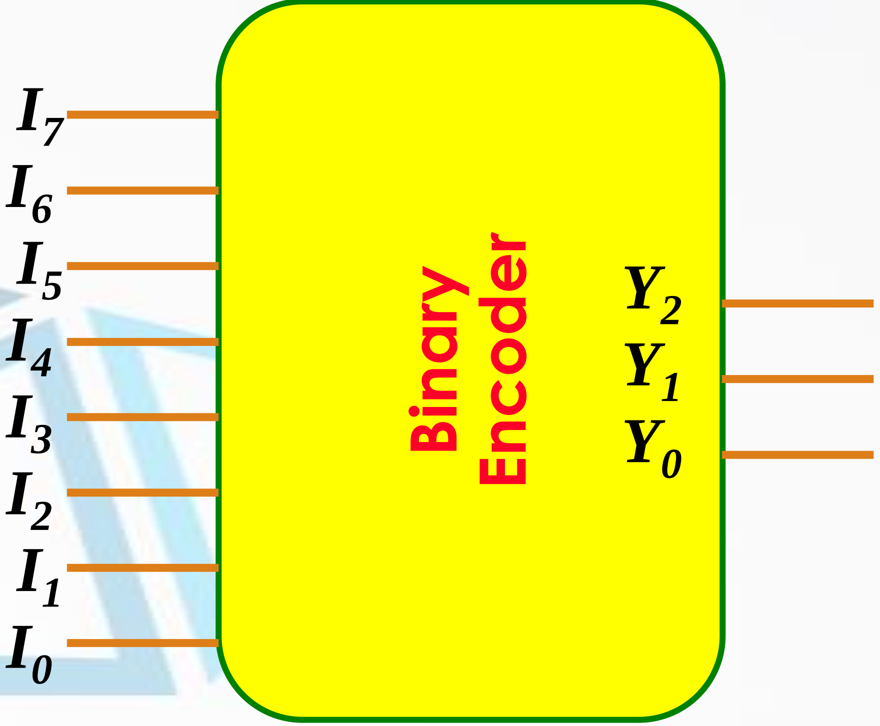
x_3	x_2	x_1	y_1	y_0
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
1	0	0	1	1

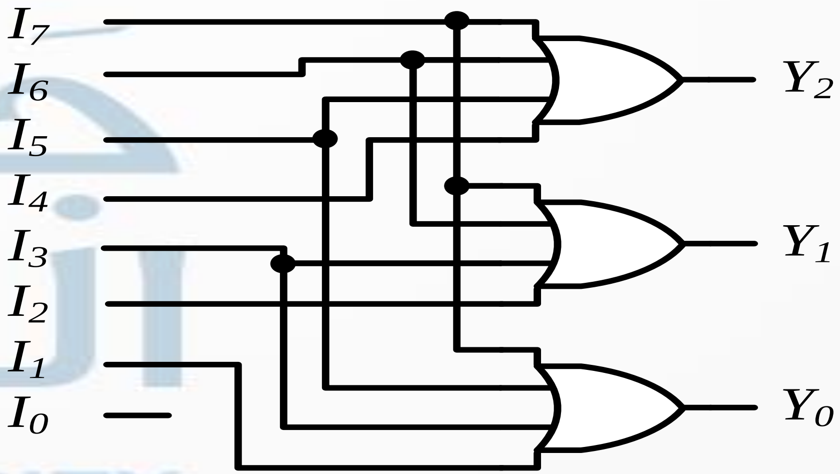
Encoders

- Octal-to-Binary Encoder (8-to-3)

I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0	Y_2	Y_1	Y_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
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

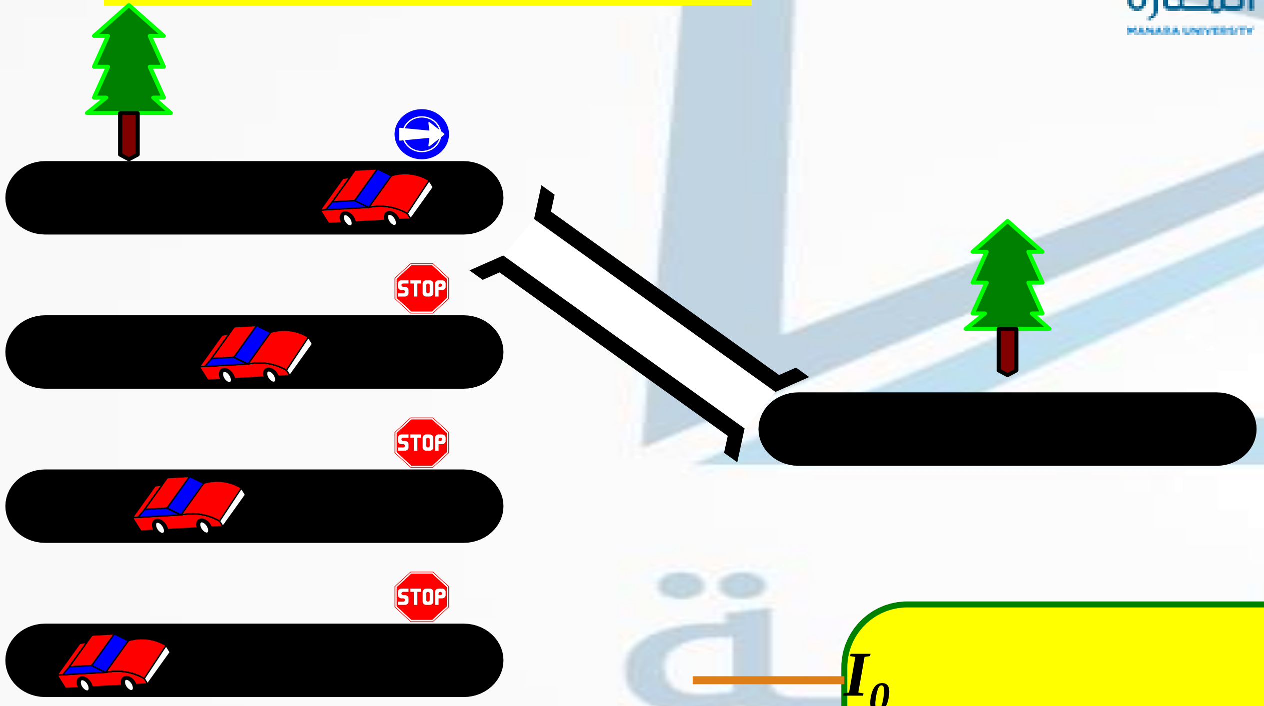
$$Y_2 = I_7 + I_6 + I_5 + I_4$$
$$Y_1 = I_7 + I_6 + I_3 + I_2$$
$$Y_0 = I_7 + I_5 + I_3 + I_1$$



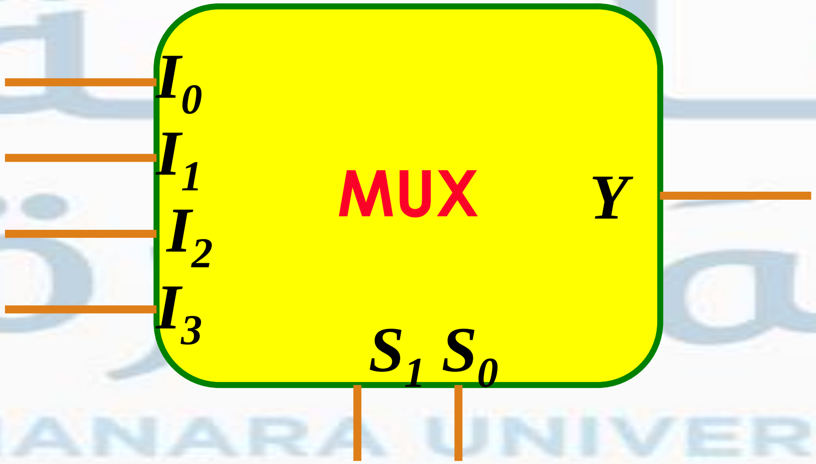


Encoder / Decoder Pairs

Multiplexers

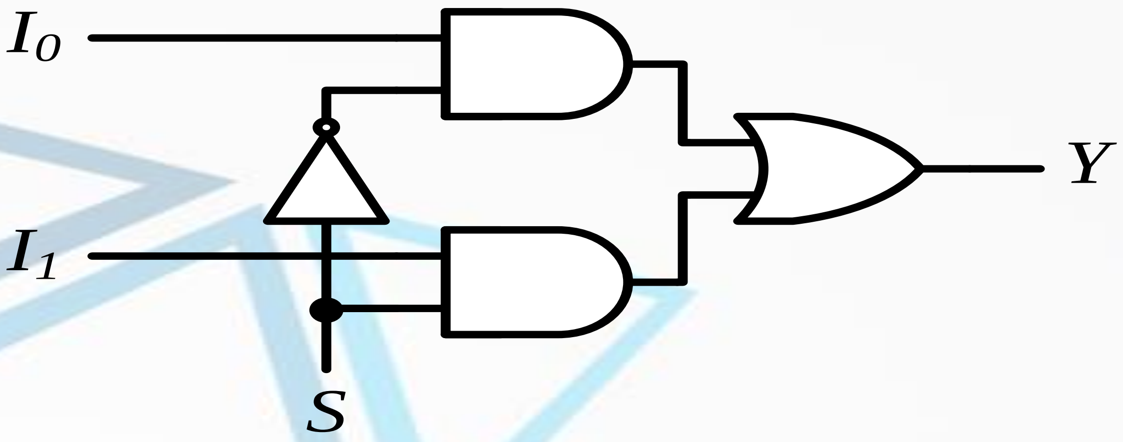


S_1 S_0	Y
0 0	I_0
0 1	I_1
1 0	I_2
1 1	I_3

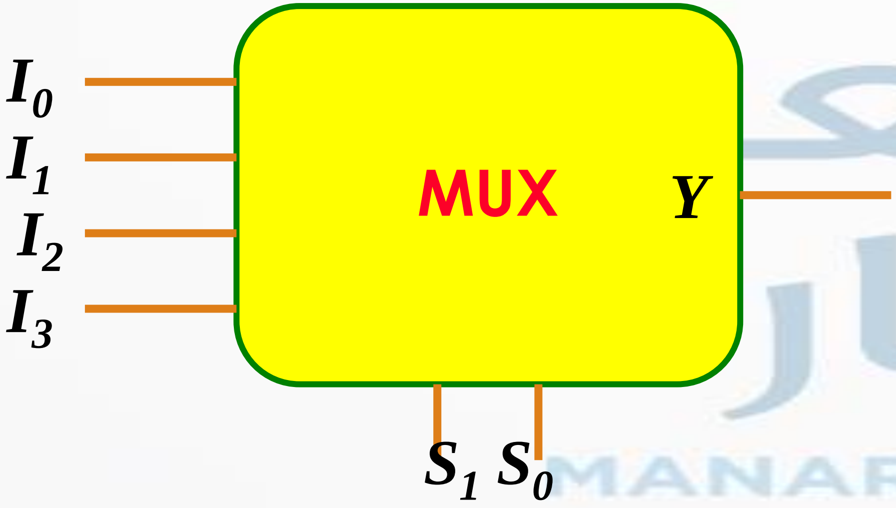


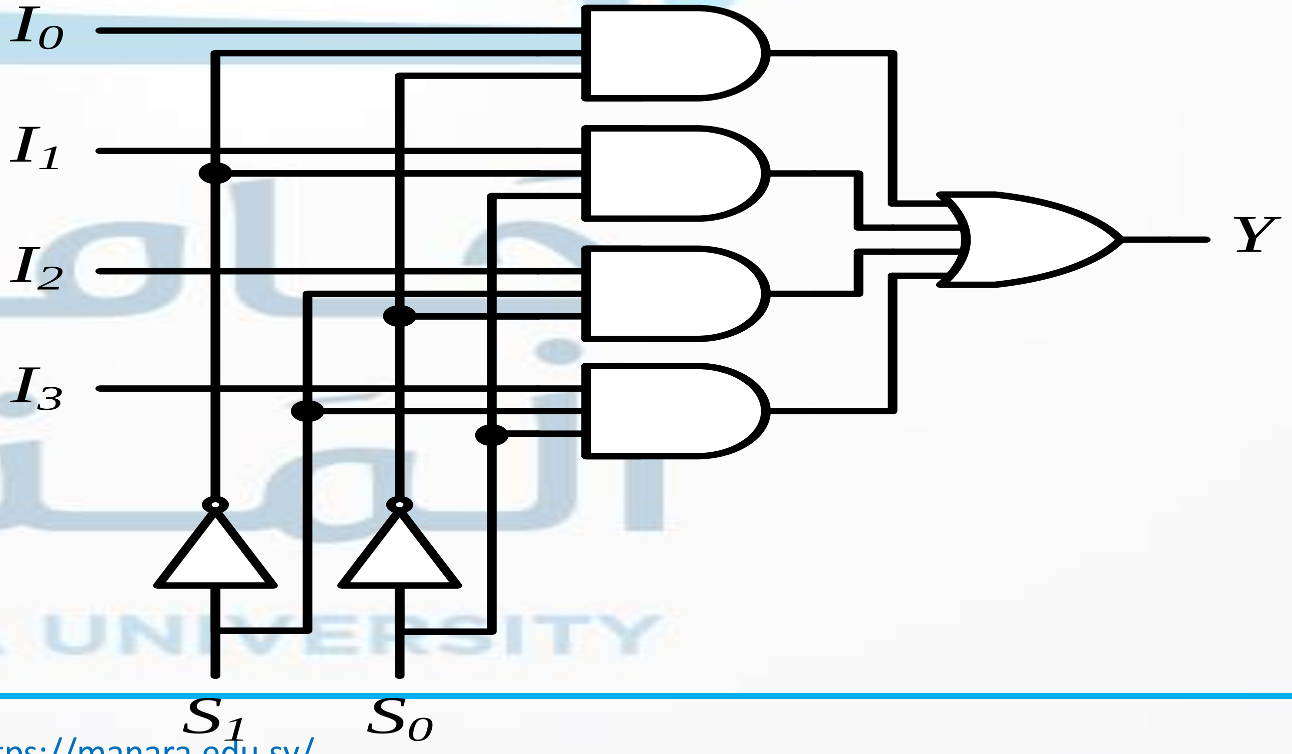
Multiplexers

- 2-to-1 MUX



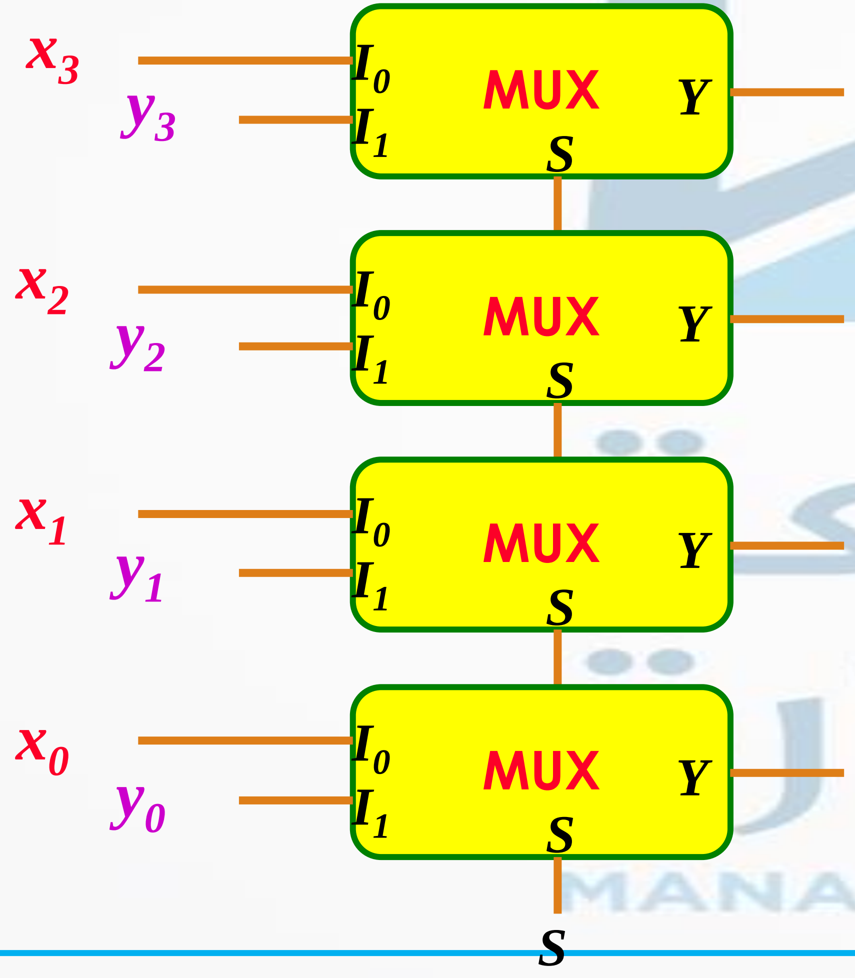
- 4-to-1 MUX

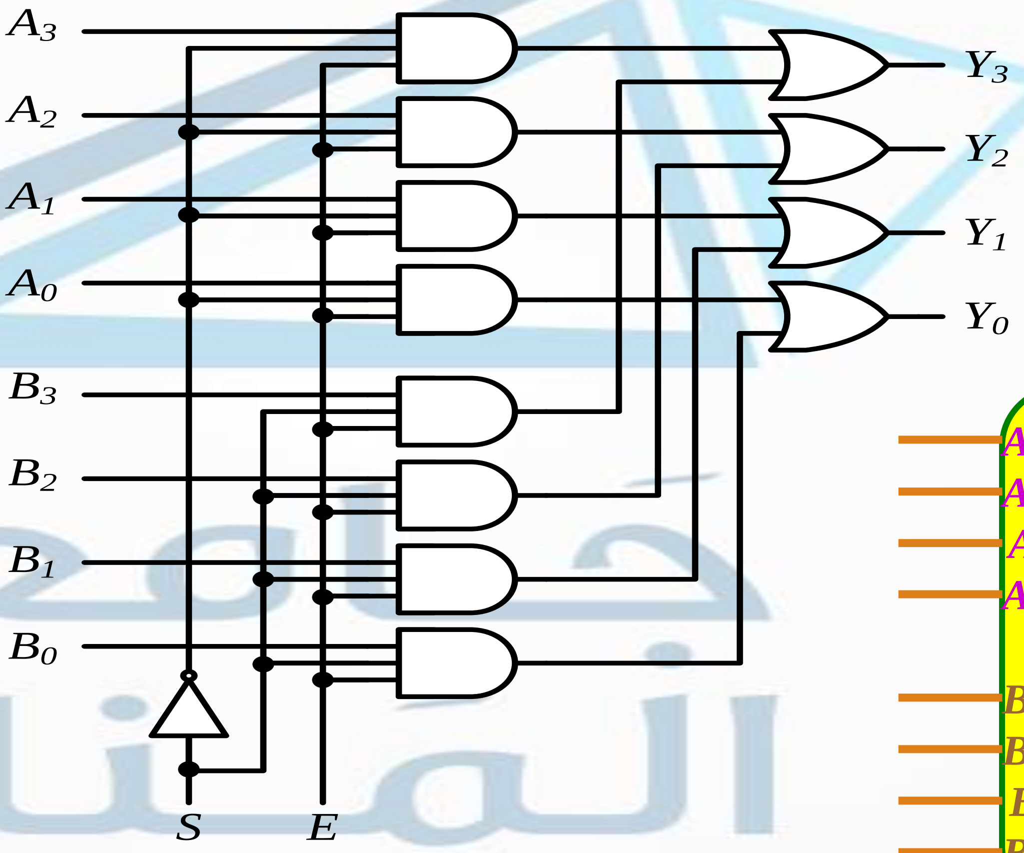


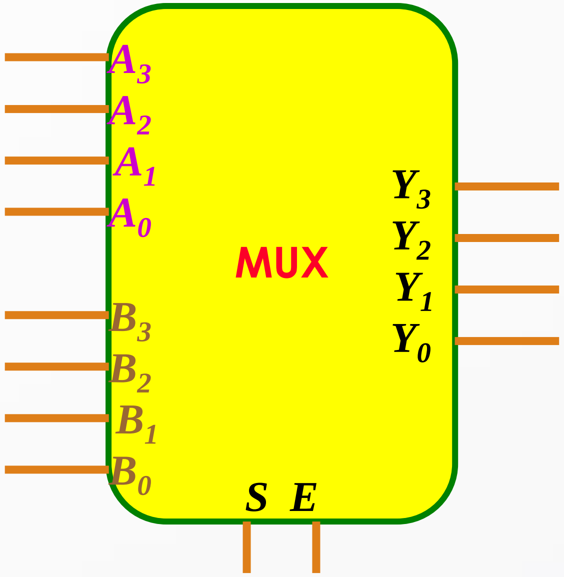


Multiplexers

- Quad 2-to-1 MUX





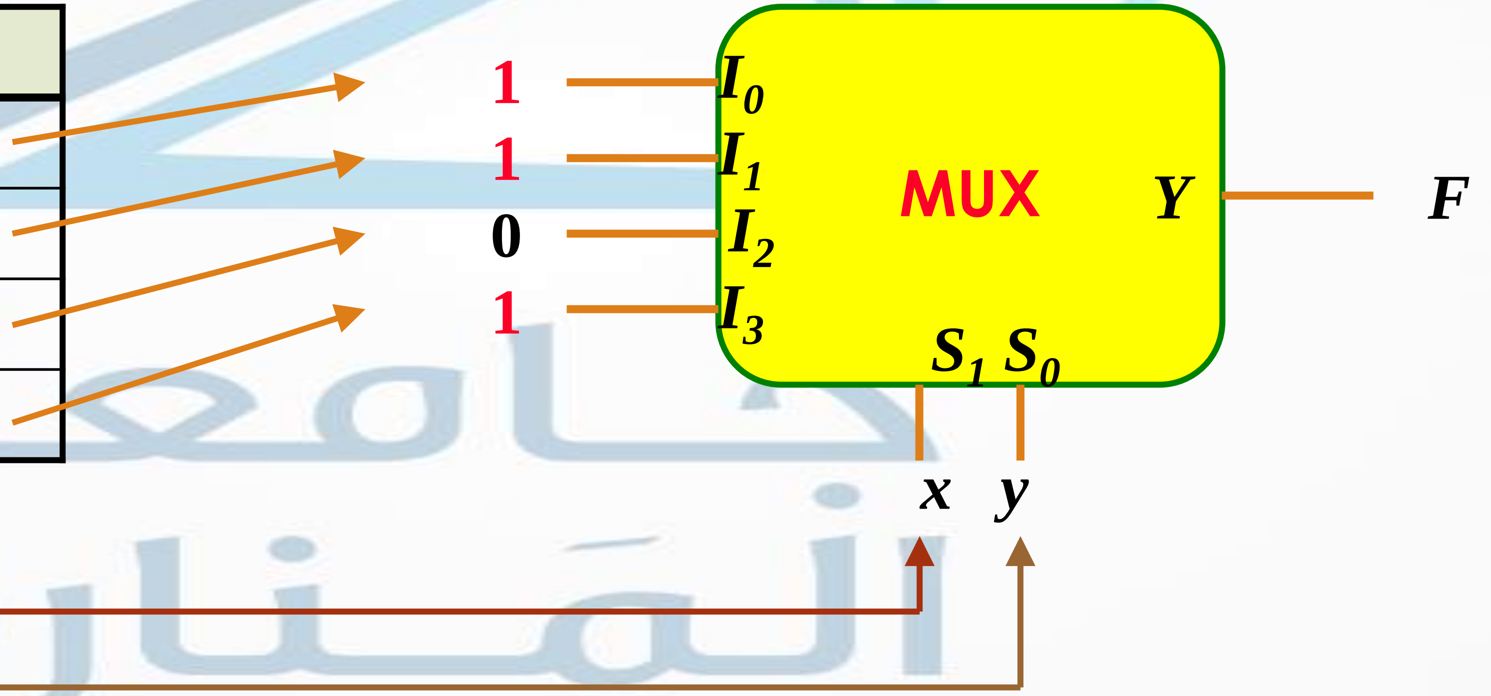


Implementation Using Multiplexers

- Example

$$F(x, y) = \sum(0, 1, 3)$$

x y	F
0 0	1
0 1	1
1 0	0
1 1	1

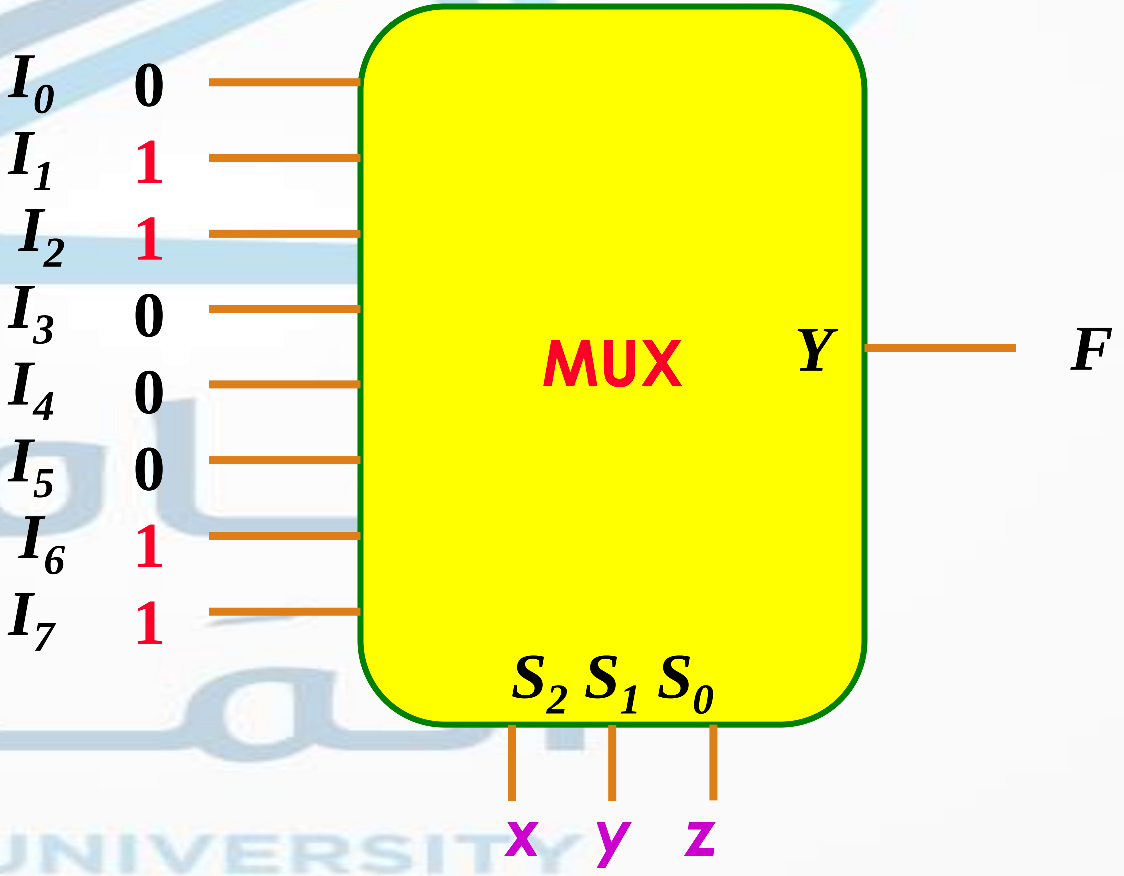


Implementation Using Multiplexers

- Example

$F(x, y, z) = \sum(1, 2, 6, 7)$

x y z	F
0 0 0	0
0 0 1	1
0 1 0	1
0 1 1	0
1 0 0	0
1 0 1	0
1 1 0	1
1 1 1	1



Implementation Using Multiplexers

- Example

$F(x, y, z) = \sum(1, 2, 6, 7)$

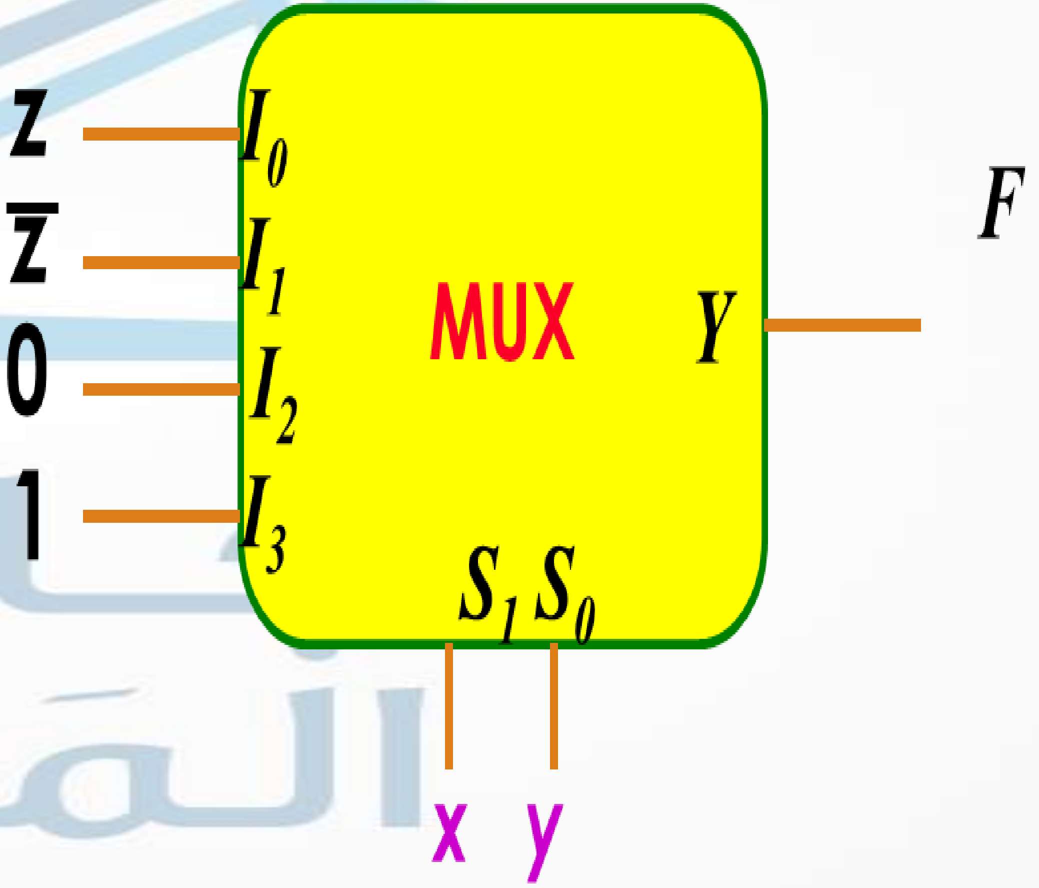
x	y	z	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

$F = z$

$F = \bar{z}$

$F = 0$

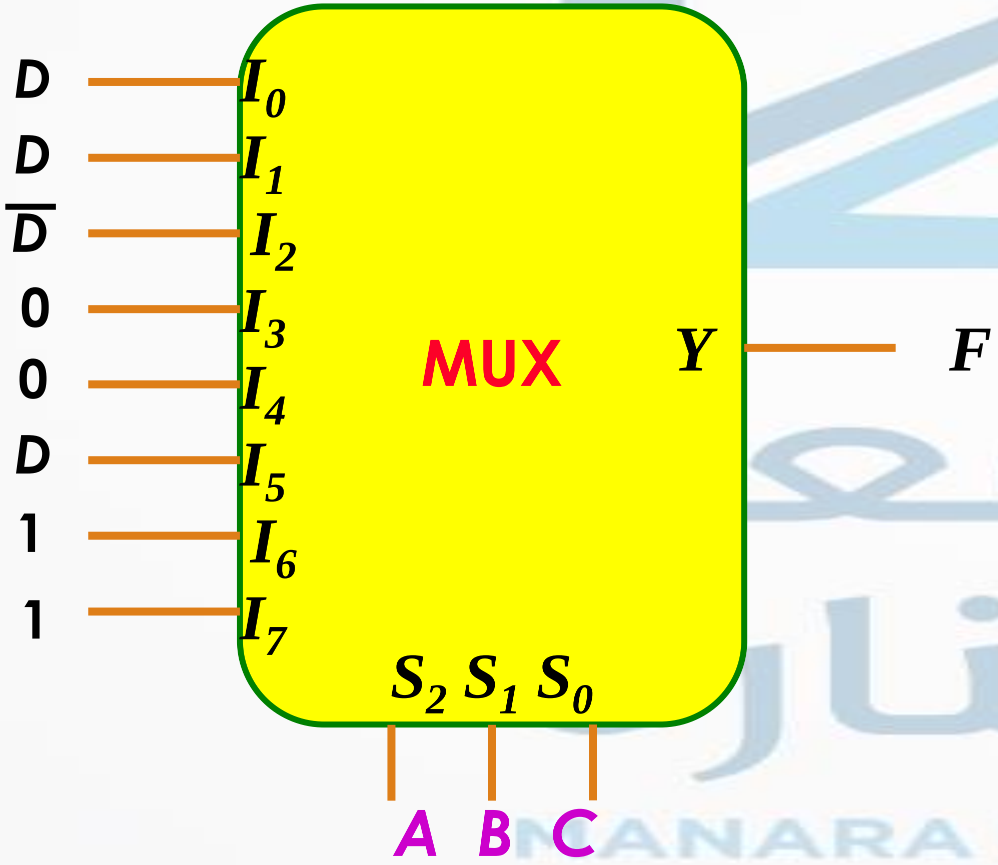
$F = 1$



Implementation Using Multiplexers

- Example

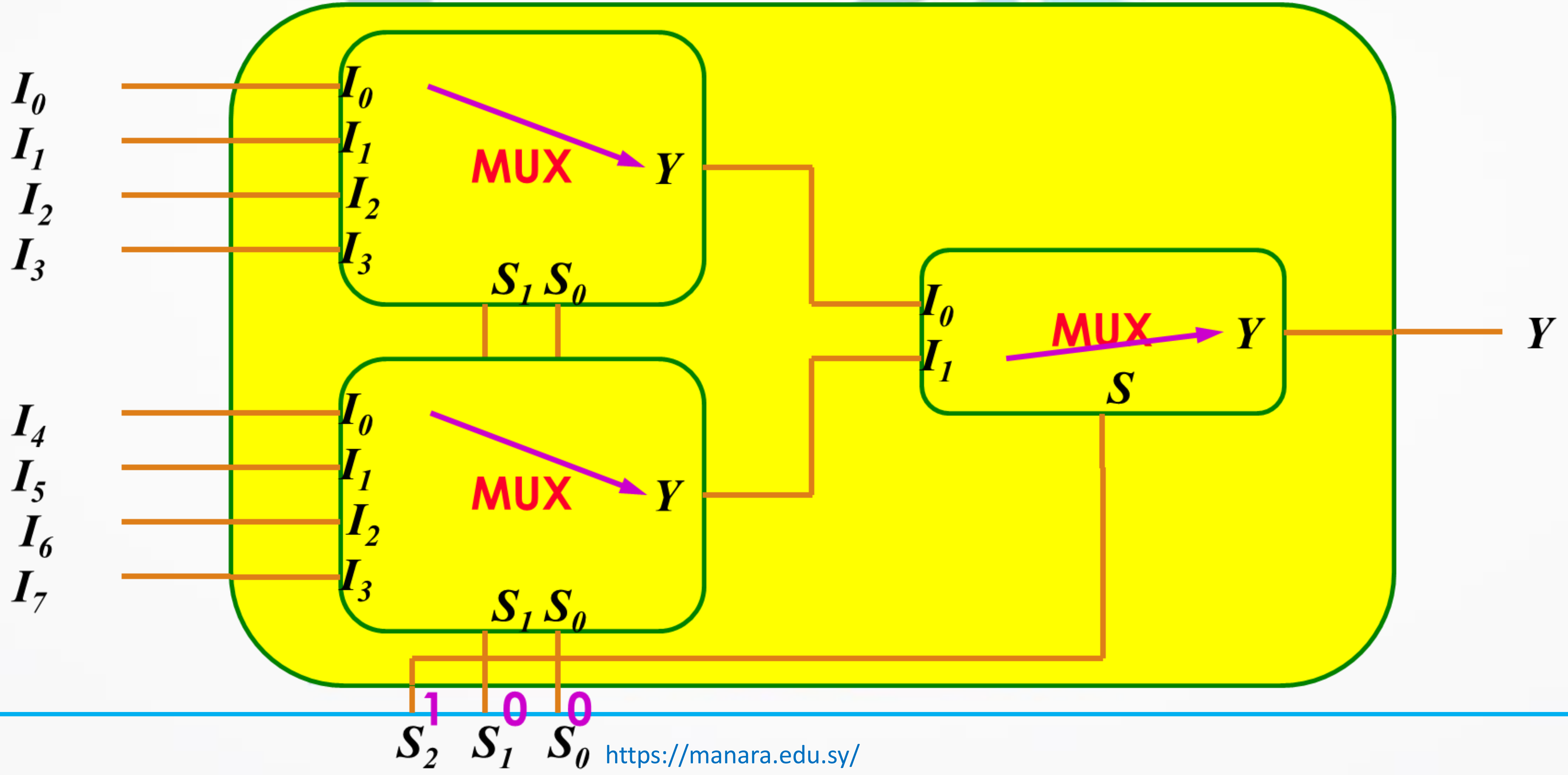
$F(A, B, C, D) = \sum(1, 3, 4, 11, 12, 13, 14, 15)$



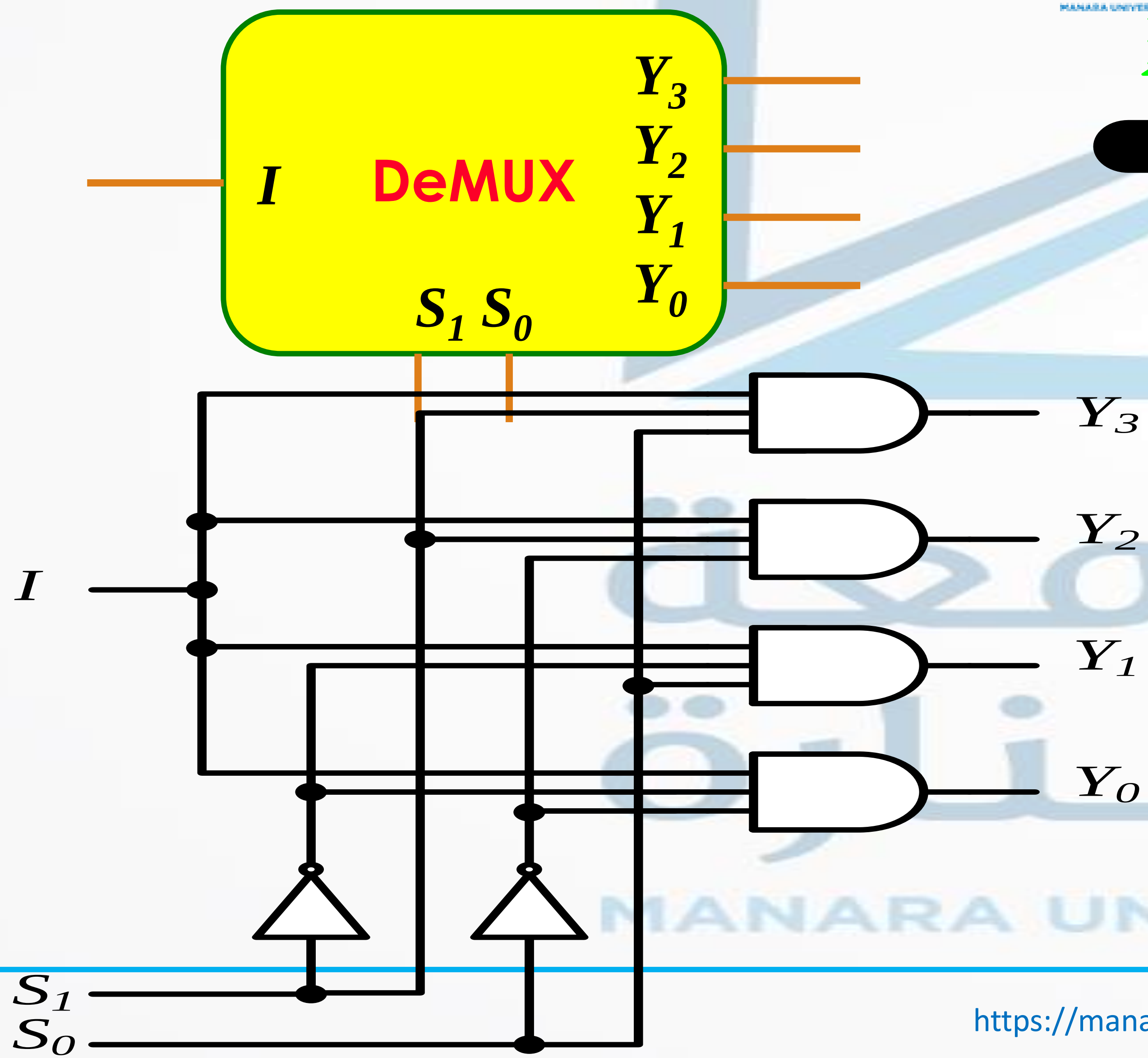
A B C D	F	
0 0 0 0	0	$F = D$
0 0 0 1	1	
0 0 1 0	0	$F = D$
0 0 1 1	1	
0 1 0 0	1	$F = \overline{D}$
0 1 0 1	0	
0 1 1 0	0	$F = 0$
0 1 1 1	0	
1 0 0 0	0	$F = 0$
1 0 0 1	0	
1 0 1 0	0	$F = D$
1 0 1 1	1	
1 1 0 0	1	$F = 1$
1 1 0 1	1	
1 1 1 0	1	$F = 1$
1 1 1 1	1	

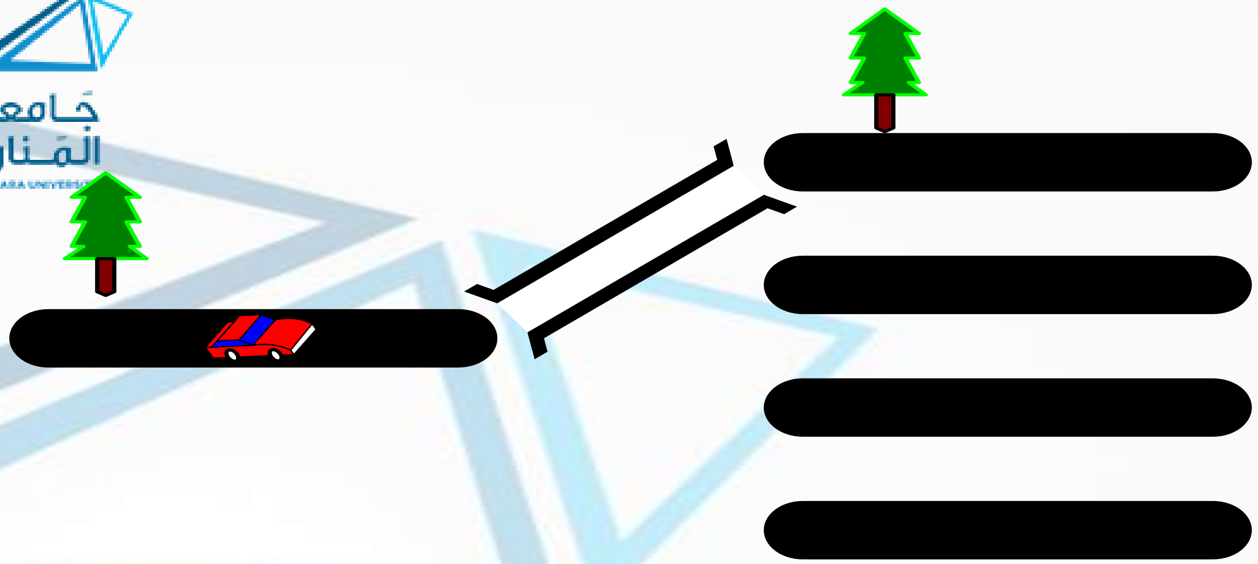
Multiplexer Expansion

- 8-to-1 MUX using Dual 4-to-1 MUX



DeMultiplexers





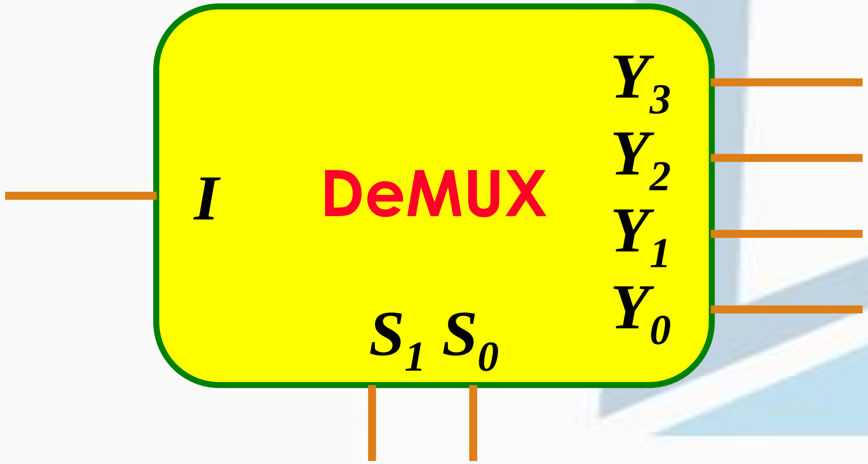
S_1	S_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	I
0	1	0	0	I	0
1	0	0	I	0	0
1	1	I	0	0	0

Multiplexer

DeMultiplexer Pairs

DeMultiplexers

Decoders

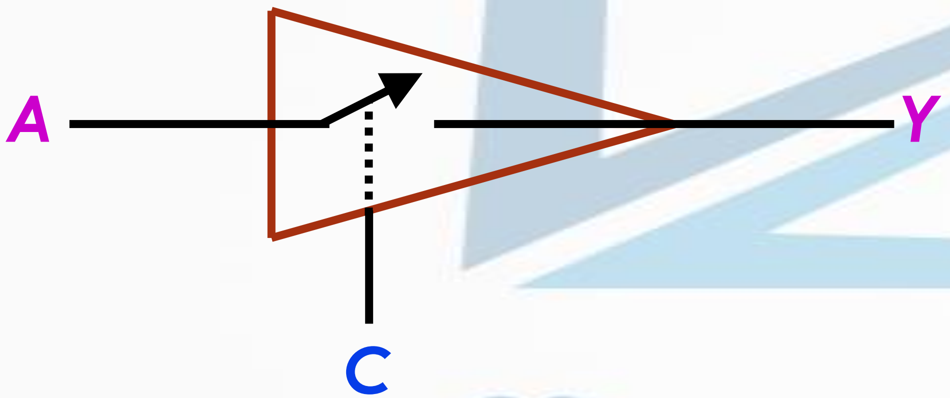


S_1 S_0	Y_3	Y_2	Y_1	Y_0
0 0	0	0	0	I
0 1	0	0	I	0
1 0	0	I	0	0
1 1	I	0	0	0

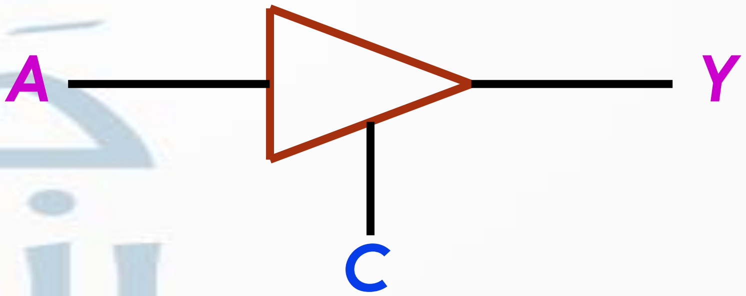
E	I_1 I_0	Y_3 Y_2 Y_1 Y_0
0	x x	0 0 0 0
1	0 0	0 0 0 1
1	0 1	0 0 1 0
1	1 0	0 1 0 0
1	1 1	1 0 0 0

Three-State Gates

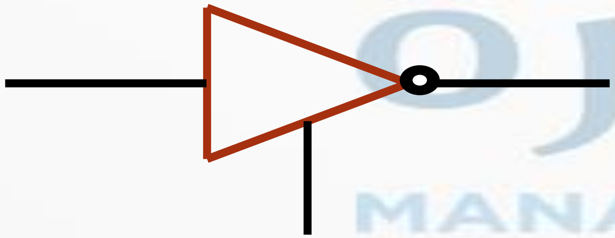
- **Tri-State Buffer**



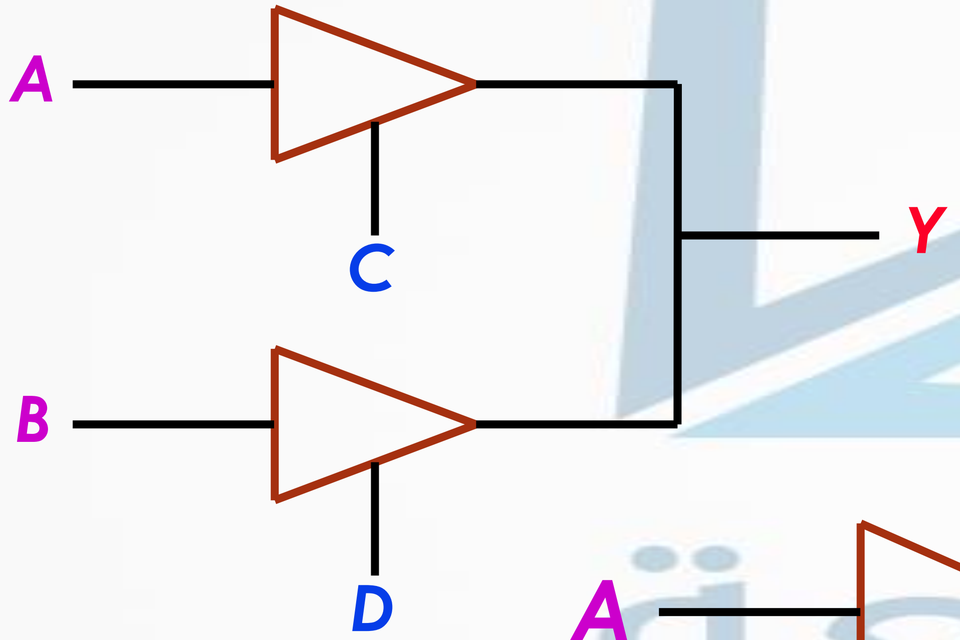
C	A	Y
0	x	Hi-Z
1	0	0
1	1	1



- **Tri-State Inverter**

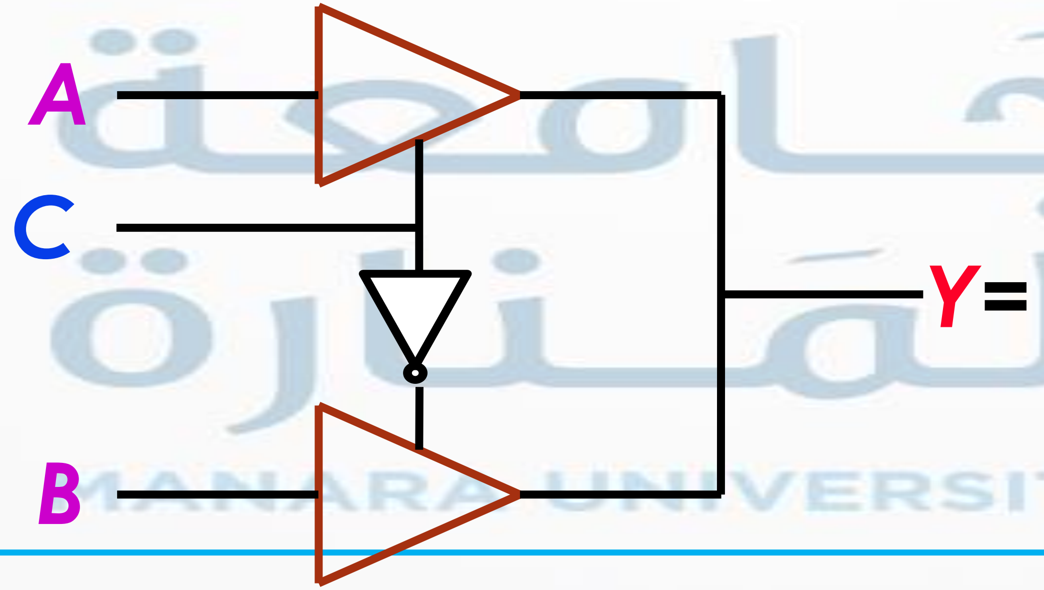


Three-State Gates



C	D	Y
0	0	Hi-Z
0	1	B
1	0	A
1	1	?

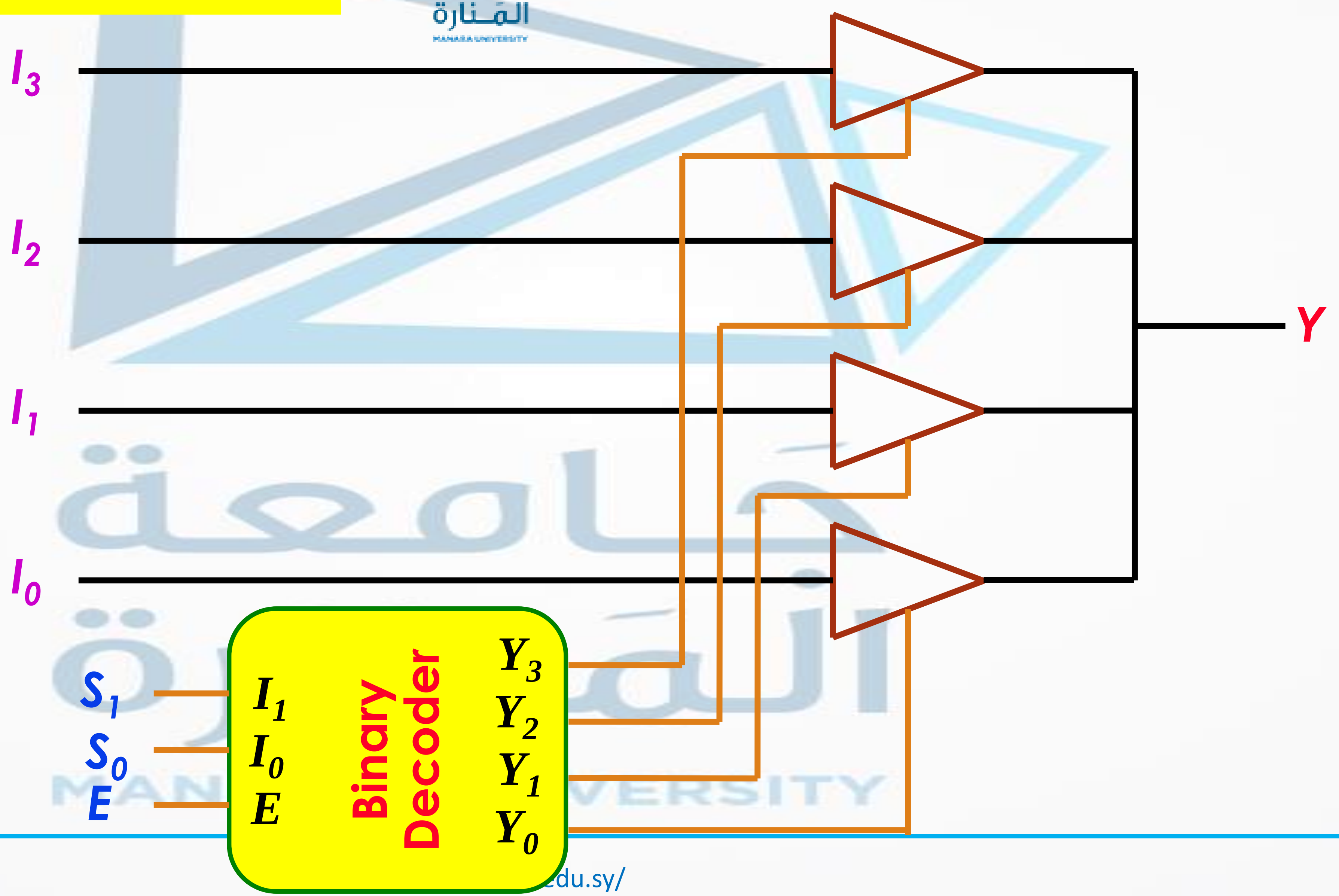
Not Allowed



A if C = 1

B if C = 0

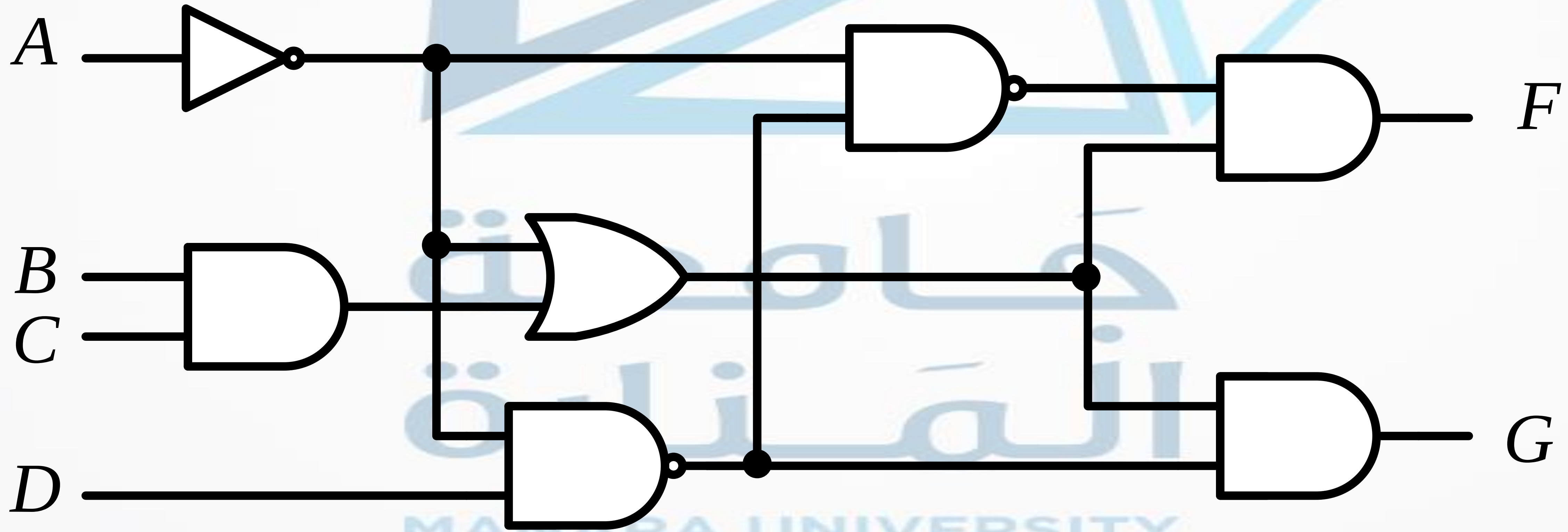
Three-State Gates



Homework

جَامعة المَنارة
MANARA UNIVERSITY

1. Obtain the simplified Boolean expressions for output *F* and *G* in terms of the input variables in the circuit:



Homework

2. For the circuit shown in the "Quad 2-to-1 MUX":

(a) Write the Boolean functions for the four outputs in terms of the input variables

(b) If the circuit is listed in a truth table, how many rows and columns would there be in the truth table?

Homework

Design a combinational circuit with three inputs, x, y, and z, and three outputs, A, B, and C. When the binary input is 0, 1, 2, or 3, the binary output is one greater than the input. When the binary input is 4, 5, 6, or 7, the binary output is one less than the input.

Homework

4-11 Design a 4-bit combinational circuit incrementer. (A circuit that adds one to a 4-bit binary number.) The circuit can be designed using four half-adders.

4-13 The adder-subtractor circuit has the following values for mode input *M* and data inputs *A* and *B*. In each case, determine the values of the four SUM outputs and the carry *C*.

	M	*A*	*B*
(a)	0	0111	0110
(b)	0	1000	1001
(c)	1	1100	1000
(d)	1	0101	1010
(e)	1	0000	0001

Homework

4-27 A combinational circuit is specified by the following three Boolean functions:

$$F_1(A, B, C) = \sum(2, 4, 7)$$

$$F_2(A, B, C) = \sum(0, 3)$$

$$F_3(A, B, C) = \sum(0, 2, 3, 4, 7)$$

Implement the circuit with a decoder constructed with NAND gates and NAND or AND gates connected to the decoder outputs. Use a block diagram for the decoder. Minimize the number of inputs in the external gates.

Homework

4-28 **A combinational circuit is defined by the following three Boolean functions:**

$$F_1 = x'y'z' + xz$$

$$F_2 = xy'z' + x'y$$

$$F_3 = x'y'z + xy$$

Design the circuit with a decoder and external gates.

4-31 **Construct a 16 × 1 multiplexer with two 8 × 1 and one 2 × 1 multiplexers. Use block diagrams.**

4-32 **Implement the following Boolean function with a multiplexer:**

$$F(A, B, C, D) = \sum(0, 1, 3, 4, 8, 9, 15)$$

Homework

4-33 **Implement a full adder with two 4×1 multiplexers:**

4-35 **Implement the following Boolean function with a 4×1 multiplexer and external gates. Connect inputs A and B to the selection lines. The input requirements for the four data lines will be a function of variables C and D. These values are obtained by expressing F as a function of C and D for each of the four cases when AB = 00, 01, 10, and 11. These functions may have to be implemented with external gates.**

$$F(A, B, C, D) = \sum(1, 3, 4, 11, 12, 13, 14, 15)$$